Deployable Prototype Documentation

(Robotic Embellishment Parking Painter)

CSUS CPE 191/EEE 193B

Senior Design Group 4.8

Absalom Yemane, Chanchiew Saeteurn, Eric Klenner, David Ryan, Ferishta Ansari

Abstract – Our team has created an autonomous line-painting robot that will save time and resources painting lines for a parking lot. This document outlines the process by which we conceptualized, designed, and ultimately constructed our device. This paper serves as a guide to the details of this project for any interested parties. The information herein can be used to inform similar projects or as the groundwork for another phase of the described system.

Index Terms—Aerosol , IMU, Encoders, GUI, LCD, Linear Actuator, UAV, IEEE

Table of Contents

Table of Tables

Table of Figures

I. Executive Summary

Our senior project chose to address the societal problem of time and resources unnecessarily spent on the task of painting lines on the ground. This field is vast and encompasses everything from sports field painting to road lines, but we chose to focus our efforts on the very specific task of painting a parking lot. The primary problems that exist in this field stem from issues with time consumption, accuracy, cost-effectiveness, and worker safety. Our solution to these problems is an autonomous line-painting robot that can perform tasks based on criteria entered through a user interface.

This robot uses multiple sensors in a local navigation system, a paint dispensation system, a robust user interface and have the ability to also employ a camera for paint detection. The overall design will cost less than $1500. It will be able to receive input from the user to determine the number and size of the parking spaces to be painted. Accurate painting is determined by accuracy within

a degree of the painted angles (usually ninety degrees) and within a couple of inches of the distance.

Our project, over the last year, took two forms. Our prototype in the Fall had the express goal of painting a soccer field. After additional testing and reviewing our time constraints, we adapted our design to paint a small parking lot instead. Additionally, our chassis was upgraded over the winter break. Our major breakthroughs came in the way of development of the chassis, navigation system, paint system, graphical user interface (GUI), and the camera system.

Each team member had a particular part of the project to work on. Chan was responsible for the chassis and encoders. Eric developed the inertial measurement unit and graphical user interface. Absalom was responsible for the sensor fusion of our local navigation system. David developed the paint dispensation system. And Ferishta worked on the camera system.

As of now the robot can navigate and paint a set path according to its programming. Issues with the navigation system prevented us from fully implementing the camera system, but it is fully functional separate from the robot.

Given the vast potential and expandability of this type of technology, we are optimistic in the viability of our project in the market. Our research informed us that we have only one competitor, the Turf Tank, and this device is only used for sports fields. That means our project has a lot of room to breathe, if we market it in the other applications. And our project could be sold at a fraction of the price.

This leads us to present the Robotic Embellishment Parking Painter Robot (REPP-Bot) as an alternative to the tedious current methods of painting parking lines. And with that being said, we are confident this technology has vast potential for further features and applications beyond what we achieved this year.

Deployable Prototype Documentation

II. Introduction

On the first day of our senior project class, we were told that the goal of the project was to find a societal problem that our unique skills as engineers could help remedy. Many ideas later, through a mutual passion for sports, we discovered that the methods by which sports fields are painted have not been updated for many years. Our initial inclination was to solve this issue for a very small market but discovered that the applications of such a technology are vast and promising. A technology that could assist with painting sports fields could also be used to mark a construction site, paint a road, or even a parking lot. Thus, our vision for an autonomous line-painting rover was born. Since then, we have developed a working rover that can paint lines on the ground for parking lots. Although our demonstration is somewhat specific, this can be further developed to include other designs and applications in the future. We will detail in this report all of the technical elements that went into making this project a success.

III. Problem Statement Overview

For CpE 190/EEE 193B, our class was tasked with tackling a societal problem via a technological solution. This task seemed daunting at first, primarily because we could not decide what exactly a reasonable societal problem would be for us to accomplish solving without getting overwhelmed. After some careful consideration and reflection, we started developing some reasonable ideas. The one that ended up taking on the most interest correlated to a problem one of our members, a sports fanatic, used to deal with growing up in sports. However, after receiving feedback from the industry last semester, we decided to automate the task of painting parking lots instead because this had a more appealing market. The issue, remaining the same, is the laborious and often, time-consuming task of painting the lines. What we realized after some further research was that this was an untapped area for automating. We decided this to be a worthy problem and moved forward to come up with this technological solution.

IV. Problem Statement

A. Time

When discussing painting lines, one of the issues that comes up is how long it takes an individual to paint perfect lines for specific dimensions, such as parking lots or traffic roads. there is a certain level of precision required when doing such a task. With road markings, although we typically do not think about it, they must be precise for the sake of safety on behalf of all drivers on the roads. Let us look at the example of a parking lot. When painting the lines of a parking lot, every spot in the lot should be evenly spaced so cars have adequate space to park. This means there is no room for miscalculations. Our society has decided the solution to this problem is to allow humans to take extended amounts of time just to make sure the paint is applied precisely to whatever job they are attempting to accomplish. However, this is problematic because the time being spent is highly unnecessary for say a robot in comparison to a human. A robot, if programmed, tested, and validated correctly can do this job efficiently in comparison to its human counterparts. Another thing to consider is the dilemma that occurs when somebody makes a mistake on their paint job and are now required to go through cleaning and reapplying the paint. In Yakima, Washington, the Sozo sports complex manager stated he spends on average 1 to 1.5 hours painting a sports field and between 11 fields, can be cumbersome to work on [1].We found when discussing with a San Joaquin road painter that they may spend 8 hours a day painting a minimum of 15 miles [2]. When it comes to sports, there is no difference. On average, football coaches at Coahulla Creek high school said spend hours weekly just painting the lines of their fields [3]. This time could be more effectively spent either discussing game strategy, at home with their family, etc.

Time is something that is invaluable, regardless of whether you're a volunteer parent or even a large construction company. For large construction company's, time is as valuable as money. Based on the Bureau of Labor Statistics, workers in the US spend about 2,080 hours annually doing paint jobs

[4]. Another fact is that workers are paid full time to paint parking lots. Based on the Bureau of Labor Statistics, these painters are being paid $18.06 per hour [18]**.** This means companies are having their people spend forty hours a week working on such jobs when their efforts could be used in other, more efficient ways. When it comes to painting lines, some companies spent several months painting parking lots of McDonalds [5]. Time is only one aspect of the entire societal problem. In the next section, we will discuss how accuracy is lost when using current methodologies of line painting.

B. Accuracy

Accuracy can be a big issue when it comes to painting lines for sports fields, parking lots, and roads. The desire for accuracy in today's rapidly developing society means we must look at the technique we implement for line painting and see how we can optimize it. This starts with first observing the techniques presently being executed. When painting lines in general, the job is usually done by a human. The problem with this approach is we have to account for human errors made on the job. One way we have mitigated these errors is by designing semi-automatic machines that need minimal assistance, in terms of manual labor. However, with that being said, the issue that arises is the machine being used is not always able to lay the paint down accurately and it having manual operations means there is still the human error factor.

Humans can only work for a certain period of time without having any errors. However, overtime, there is still a certain threshold when fatigue kicks in and causes mistakes to occur. According to a research article conducted by the *Journal of Applied Sciences*, "If an individual whose work demands have gone over the limit, it is likely that they are unable to mobilise their work effectively [6]." When a human has been overworked, the work that follows this over exertion will no longer be as accurate as when they first started. The point is to have accurate lines laid out, not somewhat accurate lines. The person doing the line painting could go faster so the time spent will not make for long hours. Painting, in general, is a task that needs time but when the person carrying out the task becomes tired, the job potentially might end up with errors regardless of how much time. According to an IEEE article by Chunfang Guo, it is said that "people in fatigue often exhibit poor physical coordination, decreased ability to dominate the brain, difficulty concentrating, memory loss, slow thinking, slow ability to judge errors, operational errors, causing an accident [9]." Guo mentions that a person who is fatigued could cost errors and have difficulty concentrating. The task of painting in general, whether it is street painting or field painting, is a task that takes hard labor. When a person becomes fatigued, paint being sprayed by that individual will not be even or parking lots that need to be a certain sizes could end up different than desired.

Fig. 1: Paint over Chalk Line
Credit:https://www.forconstructionpros.com/pavement-maintenance/article/20867977/why-to-chalk-a-parking-lot-layout

The method of designing accurate lines can also become an issue. Figure 1 above shows an image of how chalking the line first is used to make straight lines on pavement. Painting lines with accuracy on pavement can be done with many different type of methods but some can be less accurate than others. The methods can be such things as freehand painting, which can be very inaccurate because of human error. Another is string lines, which is done by laying strings done tightly before painting next to line. This alternative is time consuming. The last method is two helpers and a rope. Once again, this alternative can also very time consuming [10]. Although there are many methods to painting lines on pavement accurately, most methods take time. The most accurate and fastest way to paint lines on pavement seems to be laying down the chalk first then paint over the layout. The chalking method is

still done by a single person and still takes time to measure out the specific dimensions desired. The fact that chalking method still takes a single worker means that the worker must pay attention for hours to paint the line accurately, but the worker will become fatigue as previously mentioned. As the worker becomes fatigued, lines on pavement will not be accurate and will end up costing money to fix the mistake.

Painting could be hard sometimes when the surfaces are uneven or the nozzle of the sprayer is not able to compensate for the angle at which the paint is being sprayed. Spraying paint on uneven surfaces can cause the lines to become misshapen or crooked. According to an IEEE article, "painting a 3D model with an airbrush is a skilled discipline, fine control of nozzle speed, position and timing of the air valve are required [8]." The fact that painting a 3D model with an airbrush takes a lot of skills means that a person would have to acquire skills to become accurate and efficient at the job allocated for them to do.

C. Cost Effectiveness

The financial cost of painting a parking lot is no trivial undertaking. In some instances, a cover charge of $300 to $500 is required to just set the job in motion [11]. Once individual components of the labor are factored in the costs only get steeper. Fig. 2 is a depiction of prices to paint individual components of a parking lot from a construction company specializing in asphalt repair services.

Parking Stall (Single Line)	$4-$5
Parking Stall (Double Line With Hairpin Half Circle or cap)	$6.50-$8
Handicap Stall (Symbol With Blue Box)	$25-$35
Arrow	$10-$20
4" Line Per Lineal Foot (White, Blue and Yellow for handicap cross hatching, fire lane line, no parking zones, loading zones, bull noses etc.)	$0.20-$0.40
Stop Bar	$20-$30

Fig. 2: Parking Lot Painting Service Prices
Credit:
http://www.fixasphalt.com/blog/cost-to-line-stripe-a-parking-lot

As the figure shows, parking lot lines are charged by the line. This is on top of a blanket charge of several hundred dollars just to get the job started. Depending on the size of the parking lot, this pricing structure can start to add up very quickly.

The costs associated with hiring a parking lot painting crew with these prices are relatively small up front, but many of these factors hide in the uncertainty of human performance. One of the primary concerns any employer has is training new employees. This is crucial in line-painting, because a poorly trained employee will make mistakes that will cost time and money that could have been used elsewhere. Alan Dungey, a sales manager with Great Lakes Athletic Fields in Buffalo, New York, states explicitly that "The bottom line with painting is that you're only as good as how you're handling your machine. You have to keep your machine clean and use the right nozzles--that really affects the quality of the line. After you are finished painting, you have to be very thorough in cleaning your machine. Ninety percent of failure when it comes to painting comes from not cleaning your tank properly, so when you mix the next load in you end up with chunks of old paint that clog up your nozzle [12]." As Dungey states, many failures and setbacks occur in line-painting for very simple issues that stem from proper training and attention to detail. In fact, most of the time spent painting designs on the ground is focused on the preparatory work that is needed to measure out what must be painted [12]. Training and competency are large variables that can cost a great deal to clients. If this same job could be automated to be done without the need for training and at the cost of the materials used, it would be a step forward in making this a more cost-effective venture.

D. Safety

Directly related to painting lines on a parking lot are the hazards of faded or confusing markings. These hazards can lead to accidents. Regarding other line application scenarios, such as construction and road maintenance, safety is the greatest concern. Our goal is to substitute a robot in place of the workers in the more dangerous areas, leaving the human in the capacity of the operator. On a construction site, with all the heavy equipment and activity, the use of a robot will lower the number of injuries/fatalities on the site. "According to a study done over a 20 year period, investigating

15,000 fatalities on construction sites, they found that equipment operating in reverse were a major culprit. Half of the fatalities were caused by dump trucks [14].” These statistics are only describing fatalities and not including non-fatal injuries. Another source saw an 17% increase in construction site fatalities from 2010 to 2015 [15]. There are many instances during a construction project where outlining is needed. These scenarios include establishing perimeters and marking out parking lots. An autonomous robot replacing the personnel involved in these tasks could lower the number of accidents on a construction site and free these workers to do other tasks.

In the area of road maintenance, not only is the safety of the department of transportation workers at risk, but the average driver on the road as well. Numerous studies have shown that faded, poorly defined road markings are the cause of many auto accidents [16]. The extent of the roads to cover, both brand new and repainting, is a constant job. The crews that are tasked with this important job also work in dangerous situations. A majority of the roads they are working on are partly in use. Cones, sign trucks, highway patrol escorts, and scorpion truck mounted attenuators are examples of some the precautionary measures employed to improve the safety of painting crews as they work. Similar to the construction environments, substituting the vulnerable personnel with an autonomous painting robot would decrease the possibility of accidents. This solution also has the potential of simplifying the number of safety features needed to be employed to perform the job.

V. Requirements and Constraints

In this project, it was necessary to be mindful of the requirements and constraints for many reasons. First and foremost, the end product must set out to accomplish at least the base of what it is designed to do, leaving all extra features aside. It must be able to autonomously paint a parking lot. The remaining requirements and constraints will build upon this fundamental requirement to design a product that is economic, time-efficient, accurate, and easy to use.

The product must also be able to paint at least one lot in a single battery charge. If the robot cannot complete a single lot in a battery life, all efficiency draws to the product are effectively moot. Accomplishing the job quickly was critical to solving our problem statement, so this requirement had to be in place.

Next, the speed of the robot had to be such that it successfully painted a single lot in less time than it commonly takes a person. The robot’s speed and painting algorithm had to be designed to achieve and even best this requirement. The primary advantage of the speed of this system was the fact that it did not require prior measurement. Measuring twice to cut once is a nice saying, but a system that has the measurement done the instant it has begun cutting seems an appropriate improvement.

The system had to also complete the task to a degree of accuracy acceptable for the parking lot that was being produced. The accuracy of the system was measured in terms of the angles between the lines, the space between the lines, and the distances of the lines themselves. Only slight curvatures in the lines would be acceptable so long as a system was in place to correct curvature in real time. The fault tolerance in the distances of the lines was ninety-five percent. Angles were judged within a degree. This was not such a problem for short lines that comprise the stalls, but could be a much bigger problem for a line running across the bottom of all the stalls. Since the system had to be accurate enough to compensate for the most sensitive measurements, we said the angle must be within plus or minus one degree of a true right angle. The necessary equipment for measuring this and feeding the measurements back for testing was configured into the design.

To be economic, we wanted this design to use paint efficiently. It could not distribute paint to already painted areas or unnecessarily backtrack over already-painted areas. An algorithm was developed to paint the lot with all requisite lines covering the least amount of ground and backtracking as little as possible.

Finally, the unit had to be easily operable by a single person. The goal of the autonomous painting system was that it would make the work easier by virtue of both its software and physical interface. It needed to have a physical interface for transporting it across the ground without the need of lifting it for easy access to the paint site. The software interface

had to be simple and robust enough for a user that is not an expert.

VI. Potential Design Ideas

To meet these requirements and constraints, multiple designs were envisioned to accomplish them. This section is an overview of each of these major design ideas and the benefits and drawbacks of each design.

A. Parking Lot Paint Sprinkler System

We briefly discussed the possibility of creating an infrastructure within the lot to distribute the paint autonomously. This idea took the form of a "sprinkler-like" system in which the paint is stored in compartments underground and distributed via powerful pumps to the mechatronic nozzles throughout the field. The nozzles would have been pre-measured and carefully installed so as to paint a specific pattern and create the necessary.

There are a number of benefits to this idea. Not the least of which is that it completely bypasses all design complications of mobility. Accuracy may prove elusive with a moving robot distributing the paint. Not to mention a moving distribution system can only distribute sequentially by default. This system could theoretically distribute the paint payload in parallel cutting the time to paint the parking lot to a mere fraction of what it was. Besides being able to distribute in parallel, a stationary system with a program that does not require movement for distributing paint is measured once and never again. Unlike a mobile distribution system, this application has the potential for being finished in just a few minutes.

This design idea is not without its flaws. It is not easily scalable for different lot sizes and designs. Since the lots are measured beforehand when they are installed, they can only do one size and design of the lines. A system like this operates with the assumption that the same kind of parking lot will be painted in the exact same spot for the entire lifetime of the system. Another key drawback to this design is sheer expense in time and money that installing this infrastructure will cost. Not only that, but the maintenance of this system in cleaning paint tanks and multiple distribution nozzles is a burden even greater than the cleaning of an individual machine.

B. Unmanned Aerial Vehicle (UAV) Distribution

The next design solution considered was an unmanned aerial vehicle as seen in Figure 3 or a drone solution to the problem [22]. The attractive part of this solution lies primarily in its ability to achieve higher speeds more easily than a ground-based system since flying is inherently faster than driving in good weather conditions. This is due mostly to the fact that air resistance is much less than the friction of travelling over the ground. This enables the system to transition quickly from already painted areas to areas that need to be painted. And on that point, ground obstacles cease to be an issue for a UAV system.

Fig. 3: UAV Drone
Credit: PCMag.com

However, this design has a number of issues that would need addressing. Most notably, power distribution and weight constraints would be very large hurdles for a UAV system. Flying requires a great deal of power. Power dictates a larger power source. A larger power source adds to the weight increasing the necessary power required to keep the device airborne. These kinds of cyclical problems can cause all kinds of problems for a design team. Another weight constraint is the load of the paint in the system. Can enough paint for a sports field be carried in an aerial vehicle while maintaining the power requirements of being able to finish the field in a single battery life? This problem is vital to our design constraints and a unique challenge in an aerial distribution system. These are all things that must be considered when developing this system.

C. Rover Distribution

An autonomous rover as seen in Figure 4 was one of our more popular proposals giving us the most flexibility between size, power, and adding additional features further into the design process [19]. There are already many open source solutions to autonomous rover mobility issues that are ready for adaptation. Some of these functions include driving in a straight line, turning at an exact angle, and going a set distance This would be a strong support in the design of such a system. This system would also be able to operate in inclement weather unlike its aerial counterpart. Best of all, the design challenges of this model, have relatively inexpensive solutions making this an ideal solution.

We do have to consider, however, that this system may encounter obstacles on the field and progress on the field may be hindered. Some of these obstacles might be objects, tall grass, or even uneven field terrain. If the presence of one or more of these obstacles prohibits the task, the rover will not meet our design requirements. Also, if the rover is too heavy, repeated and lengthy contact with the grass may damage the field. But all things aside, this seems to be the most promising idea.

Fig. 4: TurfTank Rover
Credit: TurfTank.com

D. Robotic assisted Paint machine

One of the options that came up was a robot-assisted paint machine. This solution would energy efficient because the machine is not a vehicle, unlike some of the other solutions. The solution would need to be operated by a human operator but the machine will bring down the error to approximately zero. Thus, the system has now solved our problem, in terms of accuracy, which will inherently solve the issue of money since repaints because of mistakes made the first time will be reduced to zero.

While there might be benefits to this sort of design, there are definitely drawbacks. One of the first drawbacks is that this design requires there to be operated by a person, which means it is technically not autonomous anymore. This introduces back the issue of someone having to spend their time working on the field rather than the robot taking care of it for you.

E. Vehicular Distribution

Another option we briefly considered was the idea of a golf-cart like vehicle being adapted for the task. This would give us the largest power and carrying capacity. This design could do multiple fields on a single fuel tank and carry enough paint to complete multiple lots. This idea also has the flexible potential to be automated or operated by a single user. This idea has the easiest method of interaction between system and user, since driving is a common skill to find in employees.

This design idea does have its share of bugs. Because of its size, it will be expensive to build and transport. We will be adding mechanical complexities with a gas-powered engine as well as the cost of the gas itself. By and large this idea is a long-shot for the budget and scope we will be able to achieve in one year.

VII. The Competition and our solution

A. Information on Competition

After discussing the research and accounting for the associated pros and cons, we decided to go with the rover distribution system. In the course of our research, we found three other similar approaches to this type of design application.

The first was the TurfTank rover based out of Atlanta, Georgia [19]. This rover type application uses GPS and GLONAS satellites to provide guidance for the robot as it paints sports field lines. It has a compact frame design with a differential drive wheel arrangement.

The second was Fieldroid. The robot was designed by a team from Carnegie-Mellon University to paint the lines on a soccer field

specifically [20]. The team used a Leica Robot total station that uses a laser system to determine the location of their robot. the Leica system has a minimum market price of about four thousand dollars depending on the kit you purchase.

The last robot is the product of a team from the University of Waterloo [21]. They designed a small scale differential drive robot to paint lines. The device does not have any position determining equipment. Rather it uses data from the motor encoders to determine any deviation from a straight line. The system makes adjustments accordingly using an IMU and Kalman filter when deviation is detected.

B. Design Uniqueness and Value

The idea of automating the painting of lines, especially in parking lots is a fairly untapped market. The three designs listed above are the only recent documented projects we were able to find towards a simple system to perform this task on a sports field. The aspects of our proposed system that set our project apart from the other projects are in the areas of guidance, power consumption, and time. The TurfTank uses sophisticated GPS technology to determine the robot's position and path. This accounts for a substantial power draw, and it requires a paid subscription to use the satellites. The Fieldroid system uses a very high cost, industrial grade, laser detection system to determine the location of their robot. Our intention is to make a system that will not require such a high initial cost or subscriptions. The Line-It-Up robot is a simple design that uses wheel encoding to run in a straight line. It is focused on painting brand new lines, with a constraint of performing the task in under two hours. The system we developed does not require paid subscriptions of any kind. The navigation system for our robot was designed with lower cost devices that could still obtain a high level of movement accuracy when fused together. ‘ Lastly, we did not rely on just one type of device, such as the Line-It-Up robot, to nullify the possibility of any one component failure grounding our robot completely.

C. Design Solutions to Societal Problem

Our robot design addressed all the major concerns with our societal problem in the areas of time, cost, and accuracy. Its applications can be expanded to yield solutions to safety as well. The design constraint that this system must measure and paint a parking lot faster than a person can, presents an immediate solution to the time problem. Since the measuring process usually takes more than one person, and these teams are often painting more than one area, a single autonomous system will necessarily save the man-hours preparing and painting these lots. These man-hours would be replaced with the task of maintaining and monitoring the system. Thereby saving the time of these workers so that they handle other aspects of the job.

The accuracy of such a system measuring distances and angles with hardware instead of string and human accountability would address the issues of accuracy with this job. Worker fatigue and lack of training in accurate measurements would no longer be a factor. This would also minimize the need for repaints and corrections.

The cost of such a system would be lessened in the long run. The time of running it would be much less than multiple people doing the same work with minimal chance for errors. This would lessen the long term cost of painting and repainting.

If this idea is ever expanded to the field of construction, an added value will be placed on it keeping the workers safe. Since an autonomous system ensures worker safety by allowing workers to monitor the work at a distance. This aspect of our problem is solved by default. Accidents may still happen, but human injuries and fatalities will be greatly reduced by the presence of an autonomous system doing the job.

VIII. Robot Features

A. Robust User Interface

The user interface, similar to the one in Figure 5, serves the user in two important ways [24]. First, the user interface would be what turns the rover on and off. Along with that, the display would also give the user options to customize the painting job.

Namely, it should have the ability to dictate the number of parking stalls that the robot will paint as well as the type. This will depend on whether the job will be in an open space versus against a curb.

Fig. 5: Marking Machine Control Unit
Credit: industrialproductsfinder.com

B. Spray Paint Dispenser Unit

This unit is the system that is be used to deliver the painted lines on our parking lot. It is attached to the front of the robot. It is a cable activated aerosol system controlled through a relay arrangement manipulating a linear actuator. This allows the microcontroller to activate the spraying of an aerosol striping can as seen in Figure 6 [23].

Fig. 6: Aerosol Marking Paint
Credit: asphaltsealcoatingdirect.com

C. Accurate Turns

While going through the task of painting the lines of a parking lot, the rover maintains a primitive understanding of its location on the lot to ensure proper dimensioning. We implemented an IMU to our rover to achieve this. An IMU, as seen in Figure 7, is an inertial measurement unit, which is comprised of gyroscopes and accelerometers, and if done correctly, will serve as our primary navigation system [25]. It works through the use of a gyroscope, where it measure the angular velocity of our wheels, and then uses that to come up with the angular position relative to its starting point. Following this computation, the IMU then uses it accelerometers to get the reference frame it is working in, which allows us to come up with our linear position. We used rotary encoders to verify that the IMU was tracking the correct position within a certain error threshold. A rotary encoder is a piece of electronic circuitry that can be used to detect the rotation angle of the motor shaft, which can then be used to calculate an approximate position.

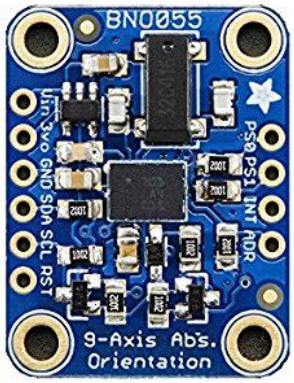


Fig. 7: Adafruit IMU
Credit: .adafruitlearningsystem.com

D. Obstacle Detection

We wanted to implement a system for physically detecting the object when it comes in front of the rover. The obstacle detection, if implemented, would be using ultrasonic sensors that detect and alert the microcontroller if there is an object in front of it at such a close proximity. If the object is a curb, the system would respond according to it's programming. If the object was an unexpected obstacle, then we would need to have an alert system to make the user aware of the issue and make a correction.

E. User Alert System

This feature is to have a system in place that alerts the user when an obstacle has gotten in front of the rover. This would allow the user to move the obstacle out of the way of the rover.

F. Distance Tracking/Speed Control

Distance tracking and speed control allows for our robot to know its true distance from the starting point and this in turn further allows us to track speed relative to its local position. The feature lets the user have an option for the length of the line painted. The necessary hardware to implement such features are encoders and motor drivers

The encoder option allows us to choose the precision for the robot. The encoder further allow the robot to track distance and speed. The encoders count the amount of ticks for each wheel. So with encoders that are more precise with each rotation the more precise the robot will be able to track distance and speed.

The choice of motors drivers were endless but the motor driver needed to be able to handle a certain load size. Figure 8 shows a motor driver that could be used to control the motors [26].

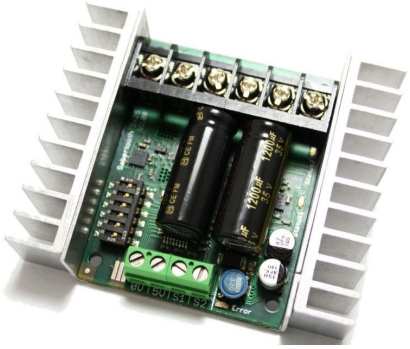


Fig 8: Sabertooth 2x25 Motor Driver
Credit: robotshop.com

F. Paint Detection

We designed a paint detection feature that was used to determine if the paint system runs out of paint during the job. This task could either be accomplished through the use of a camera or IR sensors. Using a camera would require a separate microcontroller that is capable of running openCV, which is a library of functions for machine vision/learning applications. Using IR sensors would require a very simple program, running on a microcontroller, that would adjust the wheels of the robot accordingly when the sensors detect a line. Both of these options were explored, and we ultimately went with a raspberry pi microcontroller using a pi camera.

IX. Cost Breakdown

Our project was sponsored by Intel. We had $500 in sponsorship money to put towards the project. All other costs after that were out of pocket for our team.

Item	Cost	In Use (Y/N)
First Semester		
Arduino Mega	$50	Y
AdaFruit IMU	$30	Y
AdaFruit GPS	$40	N
HM055B Compass	$15	N
Parallax Motor/Wheel Kit	$300	N
Parallax Chassis	$40	N
Batteries(2)	$50 (total)	N
Parallax Motor Driver	$24	N
Wire Harnesses	$5	Y
Power Distribution Board	$100	N
12V DC Pump	$25	N
12V DC Solenoid Valve	$16	N

3 Gallon Reservoir	$18	N
Sprayer Nozzle	$34	N
Total:	**$747 (semester total)**	**$85 (carried to next semester)**
Second Semester		
Chassis (used electric wheelchair)	$80	Y
Motors (minus brakes)	$250	Y
Tires/Wheels	$400	Y
Cytron Motor Driver	$30	Y
Raspberry Pi 3 (plus camera)	$45	Y
Batteries (2)	$75	Y
Battery Charger	$25	Y
LCD Screen	$20	Y
Paint Dispenser Unit	$100	Y
Paint Linear Actuator	$60	Y
Encoders (2)	$26	Y
Striping Paint	$25 (6-pack)	Y
Total:	**$1111.25 (semester total)**	**$1196.25 (final prototype cost)**

Table 1: Cost Breakdown Chart

X. Project Milestones

A. First Semester Chassis Build

The first milestone we reached in our project was the completion of the first build of our robot chassis during the first semester. The chassis and motor assembly came from Parallax. Once the basic frame of the robot was created, the other features such as the motion control systems and gps features could be tested on the actual system. All the other features and tests hinged on this first step.

B. Encoder and PID Systems

The successful development of an encoder system in conjunction with the integration of a PID controller was another major milestone for the project. The ability of the robot to make very precise movements in it's painting algorithm was dependent on the precision of this system. The encoders further allowed the robot to have a very precise knowledge of the distance it has traveled. All of the information was pivotal to the project's success in delivering some of the design contract features mentioned previously.

C. IMU System

The inertial measurement unit improved the precision of the robot's local positioning system. Using sensor fusion, the readings from the IMU and Encoders were used to give the robot painter a precise knowledge of it's orientation and movement when traveling and turning. The integration of the IMU system marked another level of precision. Bringing the robot to the level of precision we guaranteed in our contract.

D. Second Semester Chassis Build

The second semester chassis build gave us more real estate on the robot as well as a higher power capability. The build was based on feedback and testing of our first semester robot. The second version also allowed us to improve a number of the devices that we had implemented on the first robot. Using more precise encoders and having a bigger battery bank as examples.

E. Paint System Integration

The second semester saw the completion of the paint system implemented on the robot. The successful build of the system meant the robot was able to perform it's fundamental function, namely painting lines on the ground.

F. GUI Integration

The graphic user interface provided a higher level of interaction of the robot with the user. The display gave gave the user the ability to customize the job performed by the robot, as well as information regarding the robots status and progress. The completion of this feature marked the implementation of another design contract item.

G. Camera Development

The final milestone was the development of the machine vision camera system used to confirm that the paint system was functioning correctly. Since the paint system uses aerosol application paint, it is necessary to have an external system to monitor the painting and alert the robot if there is an issue.

XI. Work Breakdown Structure

1. User Interface
   - 1.1. Buttons and LCD assembly documentation (2 hrs)
     - 1.1.1. Test Buttons and LCD screen configuration (2hrs)
     - 1.1.2. Software package for feedback from IMU (2hrs)
   - 1.2. Software for user interface
     - 1.2.1. Software package to make button turn robot on and off and start (1hr)
     - 1.2.2. Software package for movement algorithm based on user input (20 hrs)

2. Motor Control
   - 2.1. Motor Assembly to Chassis and Controller
     - 2.1.1. Input/output pins schematic and assembly (10hrs)
     - 2.1.2. Wheel and motor configuration document (8hrs)
   - 2.2. Software for motor control
     - 2.2.1. Software package to make motor move as desired (20hrs)

3. Navigation System
   - 3.1. Parts Assembly
     - 3.1.1. IMU calibration (4hrs)
     - 3.1.2. Encoder calibration (6hrs)
   - 3.2. Software for Navigation
     - 3.2.1. Software package for encoder distance tracking (4hrs)
     - 3.2.2. IMU software package for path navigation (24hrs)
     - 3.2.3. GPS software package for global position (10hrs)
     - 3.2.4. Camera software for repaint(4hrs)

4. Efficient Power Consumption
   - 4.1. Power Design
     - 4.1.1. Voltage consumption research (6hr)
     - 4.1.2. Wire routing (2hr)
     - 4.1.3. Battery Spec (2hr)

5. Spray paint Dispensing
   - 5.1. Paint System Research (20hrs)
     - 5.1.1. Paint System
   - 5.2. Aerosol Paint System Integration
     - 5.2.1. Cable System Modification (6hrs)
     - 5.2.2. Linear Actuator Integration (4hrs)
   - 5.3. Camera Paint Detection
     - 5.3.1. Software package for paint detection (30hr)
     - 5.3.2. Software package for robot shutdown(2hr)
   - 5.4. Software to dispense paint
     - 5.4.1. Software package to dispense paint when desired (8hrs)

Table 2: Work Breakdown Structure

XII. Details of each work package

A. User interface

The user interface was relatively simple to design. In the initial stages, consisted of buttons for turning the system on and off and an LCD screen as in Figure 9 to display the feedback of distance traveled from the accelerometer [27].

Fig. 9: Parallax LCD System
Credit: https://www.parallax.com/product/27977

Eric was responsible for wiring the buttons and LCD screen to the microcontroller and writing the necessary code to process inputs and properly display data from the IMU. The push button and basic LCD code did not take more than two hours allowing one hour for troubleshooting. An additional two hours was afforded to properly configure and troubleshoot this software to receive data from the IMU and display it on the LCD screen. Two hours was afforded to properly document the assembly and configuration process of these parts In total this task took no more than six hours. As a matter of dependency, this package was not done until the IMU was properly installed and calibrated with its own software already complete.

B. Motor Control

The motor control feature was assigned to Chan who was assisted by Eric for the software portion of the feature. The feature was broken down into two tasks. The first task was motor assembly to the chassis, the hardware of which as seen in Figure 2, and connecting it to controller [28]. The second task was the software for the motor control. The tasks were then broken down into work packages. These work packages allowed us to determine the work needed to complete the tasks in order to complete the motor control feature. The first work package of task one of motor control took about five hours to complete. Chan had to take very careful notes on which input and output pins are being used and also made sure that the motor was connected correctly in order for the controller to communicate with motors. The second part of work package one was estimated to take two hours. Chan took careful measurements of the wheels circumference and compare it to amount of ticks per revolution of the wheels. The reason for this work package is that we needed this information beforehand in order to give the robot a good estimate of its location. Chan also had to test the motors to see if they are functioning properly. The importance of such work package was that we did not want to install broken motors. Chan applied power to the motors to see if the motors could handle what the mentioned specs are for this specific motor.

Fig. 10: Second Semester Chassis with Motors and Encoders
Credit: Chanchiew Saeteurn

Task two of motor control feature had only one work package. This work package took about twenty hours to implement with the assistance of Eric. The work package was to write code to make motor move as desired. The code written was able to make the motors move allowing the robot to maneuver. The robot is able to turn left and right

and also move forward and backwards. This motor software was then able to move as desired based on the needs of what the robot is commanded to do. The total amount hours for motor control feature was added up to a rough estimate of twenty-seven hours in order for the feature to be completed. After completion of twenty-seven hours we were done with the motor control feature and able to integrate all the features to make the project a success.

C. Navigation System

In this part of our work breakdown, we will be discussing how we decomposed our navigation system and the work that was involved. This task was broken down into two broad subtasks. The two subtasks were "Parts Assembly" and "Software Algorithms for Navigation". In this subsection of our main points, we had a more in-depth discussion of these two subtasks, their breakdown, and the work packages that were embedded within each of these two subtasks.

The first subtask mentioned was "Parts Assembly". This first subtask covered the fact that when building our navigation system for the deployable lab prototype, we had to integrate all of the parts that comprise our navigation system smoothly. With that being said, before we could worry about integrating all of the different components that make up the navigation system together, we had to first start off testing each of the components that comprise the navigation system and made sure they were working correctly. The components that make up our navigation system are the IMU seen in Figure 3 and a pair of encoders [29].

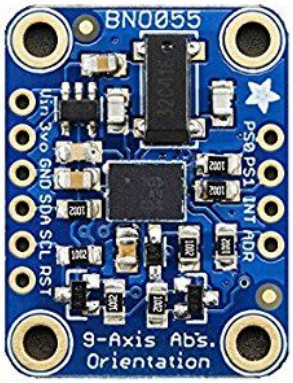


Fig. 11: AdaFruit Inertial Measurement Unit
Credit: learn.adafruit.com

Work Package one was then to make sure our first component, the inertial measurement unit (IMU), was functional and calibrated correctly. This was completed by Absalom in four hours. Work Package two was to test the encoders to make sure they were functional and check to see if their readings were accurate. Chan completed this in two hours.

The other subtask of the navigation system was the development of software algorithms for the navigation system. While it is all well and good that our different electronic components are calibrated correctly, without the appropriate software algorithms, these electronic components described in the previous paragraph are not contributing to our navigation system as a whole. With that being said, every electronic unit had its own software algorithm that needed to be implemented alongside it. These work packages are broken down based on the two sensor units being used: encoders, and inertial measurement unit (IMU). The first work package in the subtask covering "Software algorithms for navigation" discusses how encoders will be utilized and the work involved. The encoders are supposed to service our navigation system by providing the microcontroller with information on how far our rover has traveled. With that being said, work package 1 for this subtask covers the software algorithm that was developed to allow the encoder to track the distance our rover has traveled. This was accomplished in four hours by Absalom. Work package two focussed on the IMU and the software that was implemented for that unit. The IMU serves multiple purposes to our navigation system. The purposes of the IMU include making sure our rover moves in a straight path down the parking lot, helps the rover with accurate turning, and maintaining a record of the distance the rover has traveled. The software algorithm that was built for this component included functions for each of these purposes. Eric was responsible for completing this portion in twenty hours.

D. Efficient Power Consumption

This section of the breakdown covers the power system for the robot. The design and testing of the power system was pivotal for the successful operation of the robot. The first work packet for this section was the testing of the power requirements for all the electrical components. Testing their voltage and amperage usage was very important throughout the project. This packet was completed by David and Chan, and ran about six hours The next work packet will be creating and laying out the wiring harnesses for the power system to the various components being fed in the robot. Reducing clutter and loose wire will increases the aesthetics of the robot. The wiring packet was covered by Chan, with two hours allotted for completion. As components are strung together, there will likely be voltage loss. Tracking the power usages allow us to know what our overall power requirements will be to spec out sufficient batteries. The final work packet was to design a battery bank and securing it to the robot. Figure 12 shows an example of the 12V lead acid batteries we are testing with the system [30]. Chan completed this task in about three hours.

Fig. 12: Expert Power Lead Acid 12V Battery
Credit: amazon.com

E. Spray Paint Dispensing

The last section of our breakdown addressed the paint dispensing system. We began with a documentation of our research into various paint pump systems to determine which one was most likely to work with our robot system. David headed up this task, and it took twenty hours. After ordering and receiving these components, the aerosol spray system was installed on the robot. This took David six hours to complete. Afterwards, Chan and David tag teamed the linear actuator system that controlled the paint sprayer. It took about four hours to complete. The software for the paint detection system was created and tested by Ferishta. In total, the system took thirty hrs to complete. The software to shutdown the robot's movement after the camera system is not complete, but Ferishta and David have spent two hours so far. The final component of this part was be the software package designed to automate distribution of paint based on a certain set of criteria that are met. We wanted the robot to only paint when it is moving along one of the lines it knows it must paint and not at other times. The software design for this took eight hours to complete. Chan and Absalom complete this.

XIII. Risk Assessment

The following graph was generated through an analysis of the risks associated with various components of our project. Each event was given a risk factor, which is the multiplicative relationship between the impact and the probability of the event occuring. The risk scale is from zero to five with zero indicating the lowest risk and five indicating the highest risk. These risks were calculated with our early ideas on how the project would be implemented. We used this data to make informed decisions about how to continue with the project.

Risk Assessment Chart			
Name of Risk	**Impact** (levels)	**Probability** (percentage)	**Risk**
Motor Control	5	20%	1.0
PID controller	1	10%	0.1
IMU: Improper Calibration	1	30%	0.3

IMU: Misalignment	2	5%	0.1
IMU: Bias	2	50%	1.0
IMU: Timing and Noise	2	20%	0.4
GPS: Improper Calibration	2	50%	1.0
GPS: Maintaining Continuous Location Reference	3	60%	1.8
Camera: Memory Usage	1	25%	0.25
Camera: Lack of Experience	2	50%	1.0
Battery Failure	5	20%	1.0
Improper Wiring	2	10%	0.2
Over/Under Powering	4	5%	0.2
Insufficient Pressure	3	20%	0.6
Inadequate Reservoir	4	5%	0.2
Line Clog	2	5%	0.1

Table 3: Risk Assessment Chart

A. Camera System

In this section of our risk assessment report, we are going to outline some of the risks related to our camera system. For our lab prototype, the camera system does not play as impactful of a role because all it is doing is taking a picture of the line that will be cleaned up. This will provide the robot with a primitive visual of the ground and eventually acts as its eyes. So, with that being said, we will begin a discussion of the different risks involved when developing an image processing system.

1. The requirement of Memory

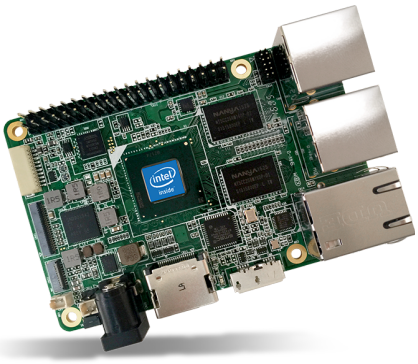


Fig. 13: UP board by Intel
Credit: UP, Bridge the Gap
http://www.up-board.org/up/

The next section outlines the risk that we had using the R200 camera. One major risk of using the R200 camera was the fact that it required so much processing power relative to our microcontroller. While we had a separate processor for the camera, this still posed a risk If the processor for the camera ran into any issues such as dying from low battery or running out of memory, this meant our camera would not be able to act as our image processor for the robot. We lessened this risk by using the UP processor depicted above in Figure 13 [42]. This processor had more memory space than specifications say are required for our image processing system to work and is used solely by our camera. With this cushion of memory for our image processing, we mitigated the risks previously stated.

2. Lack of knowledge in Machine Vision

Fig. 14: Question Mark
Credit: Steve MacDonald
http://granitegrok.com/blog/2013/07/republican-vote-stealer/attachment/question-mark

The picture illustrated above may seem abstract but depicts one of the larger uncertainties we had with our camera system. This uncertainty for us that comes with using a camera in our robot system was the fact that all of our team was inexperienced in machine vision. This lead to many mistakes down the road that we had to accept since we specified that we would be using a machine vision application in our system.

B. IMU Failures

Our longest critical path contains the navigation system for our line-painting robot. Put plainly, if this system does not function, our project has been a complete failure. A critical part of this system was the inertial measurement unit (IMU).

Fig. 15: IMU Orthogonal Linear and Angular Axes
Credit: Novatel
https://www.novatel.com/assets/Documents/Bulletins/APN064.pdf

As seen in Figure 15, this device records directional and angular velocity in three dimensions. This unit is being used as the primary measuring tool for recording the robot's turn angle. It is additionally being used as a verification for direction. If the IMU were being used as a sole means of direction thus making it a single point of failure, any false readings would have a very high impact on the execution of our project. Here we are going to take a look at some of the risks that come into play with the use of IMU devices. We did our utmost through our research to estimate their likelihood and potential impact to our project.

1. Improper calibration

Even though this risk has a fairly easy solution, we thought to include it because of how common it is in IMU systems [31]. Improper calibration is most often caused by a bad mounting of the IMU on the device. If the axes of the device are not exactly vertical and horizontal, this can lead to misreadings on the IMU that will be compounded over distance. If we did not realize that the IMU had been calibrated improperly, we could have spend many hours trying to understand why our readings seem to never work and our robot does not go straight. To mitigate this risk, we ran tests on the IMU independently to determine what should be the appropriate calibration readings and used these as a basis for all future tests. Doing this saved us potentially many hours working with no progress.

2. Misalignment

We also had to consider improper manufacturing alignment of the axes so that they are not orthogonal [31]. In the case of a non-orthogonal alignment, this is a defect in the manufacturing of the IMU, and this error will always be present. If this happens, we most likely would have needed to purchase a new IMU. The reason this could have been difficult to mitigate is that no other component on our robot is directly capable of measuring turn angles. This was a very important part of our platform, so we could not make up the difference with the other components of our navigation system. Fortunately, this situation is usually quite unlikely and we have no reason to believe it was a serious issue for our project.

3. Bias

Another risk in using IMUs is their inherent bias, which comes in the form of an on/off bias and a running bias [31]. The on/off bias is an offset that the IMU acquires each time it is turned on. If not taken into account this bias can drive the navigation into a drift that accumulates with time. One way to curb this risk is to take additional time with the IMU by itself and determine if it has biases and how

consistent these biases are. Averaging these biases will give a good bias estimate that can be factored into the configuration registers that the IMU builds into the system for this purpose [33]. To deal with running bias, we had no way of predicting how much running bias will occur because it is directly dependent on many variables such as temperature, disturbances, and mechanical stress. For this issue, we needed the other components to bridge the gap in performance. It is known that using a single IMU for navigation is frequently the cause of catastrophic accidents in spacecraft. This leads many engineers to use redundant arrays of devices used for the same navigation feature so that errors in any one device could be diminished by averaging them against many devices and accounting for variance [32]. So we turned to our encoders to correct any accrued directional biases at runtime that cannot be helped.

4. Timing and Noise

IMU units sample the angular and linear velocity at a particular frequency. Our particular unit has a maximum refresh rate of 952 Hertz [33]. Since our robot will not be moving any faster than about 3-5 miles per hour, this is not a problem getting accurate readings. This becomes a bigger problem when there is a great deal of disturbance on the ground causing vibrations in the device. These vibrations and sudden jerks produce noise in the system. This noise may appear at a frequency above the refresh rate causing small accelerations and turns to go uncalculated leading to an accumulation of error. Our solution for this problem was similar to others in that we incorporated other navigation tools with the IMU such as encoders and a PID controller. These devices all checking each other minimized the chances that stray noise particular to a single instrument misdirected the course of the robot. We could have used this cross-checking to find a scale factor that estimates the average noise across the systems and subtract these values from our instructions to our motors to maintain a clean path. However, we later upgraded our IMU to a unit that regulated its own noise and took care of this issue.

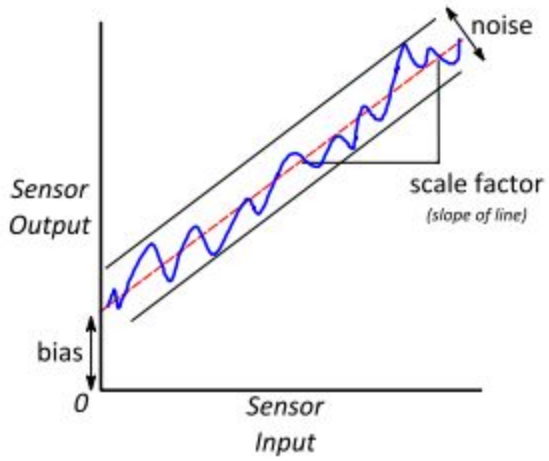


Fig. 16: Averaging Noise in a Sensor System
Credit: https://www.novatel.com/assets/Documents/Bulletins/APN064.pdf

Figure 3 demonstrates how to find a scale factor in a noisy IMU. The slope of this line becomes a scalar value that can be subtracted from the values we are reading to eliminate it from the readings. The chances of turbulence on the ground causing enough disturbance to cause lapses in calculations in our IMU due to vibration is directly related to the condition of the terrain it is operating on. To simplify these calculations, we will specify that this robot should only be run on reasonably level ground.

C. PID Issues

The PID (proportional integral derivative) being used in our line painting robot is being implemented by software. The PID software allows for our robot to detect disturbance on the wheels and then corrects for the disturbance. The use of the PID on our robot allows our robot to go straight based on the amount of encoder ticks from each wheel. The PID does this by keeping track of how much error is from one another and applies the error on to the wheel trying to follow. This of course comes with risk because what if we get the wrong readings.

1. Risk of wrong readings

The probability of the PID software failing is based on the readings coming in from the encoder sensors. If the readings from the encoder sensors are wrong then the PID software becomes useless.

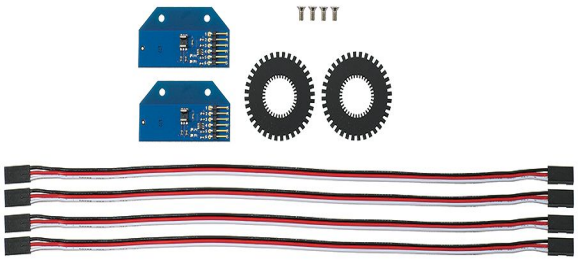


Fig. 17: Encoders by Parallax
Credit: Parallax Inc.
https://www.parallax.com/product/29321

Seen in Figure 6 above, these are the encoders that are being used in our robot. The percentage of the encoders failing are not really mention which led us to assume that the encoders do not fail as often. The low percentage of the encoders failing would not matter as much because if the encoders do fail the impact due to the failure is minimal.

2. Mitigating the PID Risk

In order for our project to proceed along we will have to mitigate the risk.. If such a failure was to happen the encoders will have to be changed out. so in order for us to mitigate the risk we must buy extra encoders for back up. This mitigation will allow us to proceed with little impact to our project because the encoders are easily replaceable and cheap. The risk is also low for our PID which means we can ignore the risk.

3. Motor failure

The motors being used on our robot are going to be DC brush motors. With the use of DC brush motors the risk is moderate because inside the DC brush motors are components that will wear down. The reason for the such high impact is because if the brushes inside the DC motors were to wear down our project will be at a halt.

As seen in figure 7 below, are the motors we are using for our autonomous line painting robot. These motors cause about $300 dollars as a pair. This is the reason for such high impact if the motors were to fail. We would then have to come up with the funds to buy another pair. We would also have to wait which cuts away at our time to complete our project.

Fig. 18: DC motors by Parallax
Credit: Parallax Inc.
https://www.parallax.com/product/28962

4. Probability of failure

The probability of the motors failing is what will dictate whether or not our risk is either moderate or high. Again the fact that these are brush motors can cause the probability to be high or low depending on the load and usage of the motor. The probability of the motors failing is moderate because it takes a good amount of usage and load for the brush components to wear down. Because the probability of the motors failing is moderate the risk then becomes pretty high because of the impact that the failure would cause.

5. Mitigating the Risk for motors

In order to mitigate the risk of the motor failure we would have to make sure that we are not applying to much load to the motors and also only use the motors when necessary. The necessary times to use the motors will only be for testing on a field and testing whether or not our motors are doing what is commanded to do with no load.. Also in order to mitigate the risk even farther we will only we testing with a specific load and never exceeding that load which gives our motors more life. These mitigations makes for lower probability of failure

which causes our risk to become lower if we follow the mitigations set in place.

D. Electrical Systems Disasters

The electrical system comprises half of this project. All the mechanical devices on the robot are controlled by electrical devices and controllers. So failure in the electrical delivery and control system would bring the robot to a grinding halt.

1. Battery Failure

The energy for the robot is supplied by battery in order to make this a mobile system. The failure of the battery system would temporarily halt the movement of the robot until we were able to replace the battery bank. There are two possibilities of failure. The first is a buying a defective battery. The prevention of this possibility would be to test the battery before purchase. The second possibility is through user misuse of the battery. For our robot, we are using lead acid batteries. Improper use and charging techniques could cause sulfation in this style of battery [39]. The likelihood of user instigated battery failure is reduced by increased knowledge of proper care and charging procedures. Figure 8 shows the recommended three stage charging cycle. In the event of battery failure, getting another battery is not expensive fix.

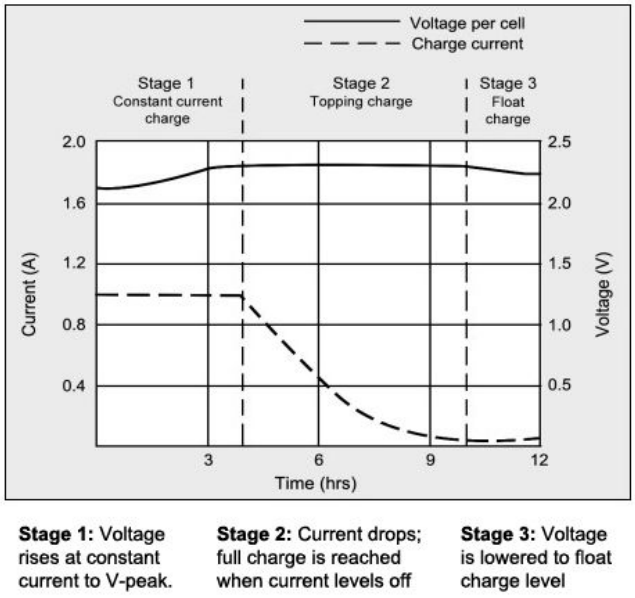


Fig. 19: Charging Cycle
Credit: Battery University
http://batteryuniversity.com/learn/article/charging_the_lead_acid_battery

Having a spare battery would minimize the time lost in work.

2. Improper Wiring

When creating the electrical system for our bot, another hazard would be to incorrectly wire up the components. There are a number of hazards involved in this [40]. Using the incorrect gauge of wire for the volt/amp requirement is one way. Incorrectly grounding the components is another. Leaving exposed wires breaks in the insulation of the wires is another type of wiring hazard. Testing of voltage and current requirements of components before they are added to the robot is one way of preventing this issue. Otherwise it will require rewiring and replacement of any damaged components.

3. Over/Under Powering

Another way of disabling the electrical system would be to have insufficient or too much voltage fed into the system [41]. Both cases cause overheating in the system. Under powering the system is especially bad for motors, because it causes the motor to draw more current to meet the power requirement. The highest probability of this event occurring is during the integration of the various components onto the robot. This situation could be prevented by thorough testing of components power requirements in operation before integrating.

E. Paint System Catastrophes

The paint system is an integral part of the purpose for this robot. It is fairly isolated from the rest of the system. The benefit of this means that the rest of robot components are not dependant on the paint system functioning. The following are the foreseeable potential issues with the system.

1. Insufficient Pressure

The system we are building for our robot delivers the paint through pressure from a pump. The pump we are testing has a ⅜ output opening. The sprayer system has a ⅞ input opening. It is possible that

there will be a loss of pressure through the at the sprayer due to the sizing difference. The main fix would be to use a different pump. To prevent this possibility, it will be necessary to test the pressure capabilities of the pump.

2. Inadequate Paint Reservoir

Another risk associated to this system would be an insufficient paint supply to paint the field. It is possible the reservoir we chose will not be able to hold enough paint to perform the job. Testing the paint requirements for our job beforehand can lower the chance of this occurring.

3. Line Clogs

Since this is a system using paint, the possibility of it developing clogs in the lines is high. If paint residue is allowed to harden in the sprayer or lines, it could cause clogs in the distribution system. if this occurs, on one side it would require some in depth cleaning to clear the clogs. On the other side it might become necessary to replace the affected items. Putting the paint system through a cleaning sequence after each use should keep this issue from becoming a problem.

XVI. Project Overview

After considering the many risks associated with the construction of this prototype, we fused many of our ideas together into a concrete project. Essential to the overall design of this robot was the use of wheels for transportation. We concluded that an aerial distribution or a manned vehicle was not cost effective enough or even within the scope of our abilities. This dictated that we find and chassis and wheels big enough to deal with potentially difficult terrain and with a load capacity to carry enough paint and and the large chassis long distances. We used a modified wheelchair chassis for the frame and tires large enough to prevent slippage on turbulent surfaces.

For our navigation system, we used sensor fusion to achieve accuracy within our required thresholds of one degree for angles and a couple of inches for distance. Sensor fusion made it possible to achieve a high degree of accuracy while not putting tremendous strain on our budget. This included using software PID controllers for encoders and an IMU. The encoders were the closest to the motors and so responsible for the lowest-level corrections to keep the rover going in a straight path. The IMU is responsible for the turns and correcting drift otherwise unnoticeable to the encoders. With an established hierarchy based on the scopes each instrument can see at, we achieved our desired level of accuracy.

Our final paint system is an aerosol system consisting of a can of striping paint that is controlled by the robot. This system turns on and off as dictated by the design of the job and has the benefit of being simple to change out. It also eliminated the risks related to pressure and inadequate system cleaning, discussed in the previous paragraphs. Its functionality can be verified by the use of an independently-developed camera system.

Our graphical user interface is a simple keypad with an LCD to give and receive information from the user. This enables the user to select the number and size of parking stall designs to be painted.

XVII. Deployable Prototype Status

As of this point, the Robotic Embellishment Parking Painter, as seen in Figure 20, is has all but one of the features fully implemented onto the robot.

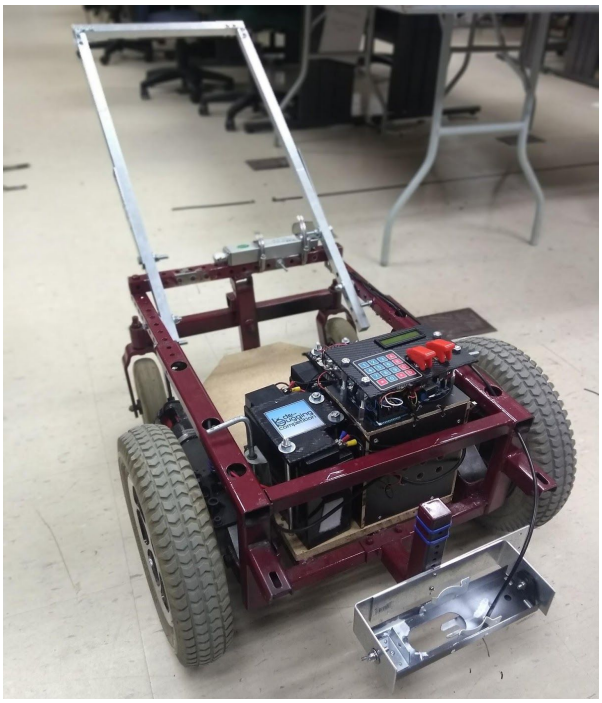


Fig. 20: Complete REPP-bot
Credit: David Ryan

The robot has a working local navigation system consisting of encoders and IMU working together to keep the robot on a straight path with the help of a PID controller. There is a functioning paint system, controlled by a linear actuator that is activated by a collection of relays to reverse to polarity. There is an input system with a membrane keypad and LCD display that engages the user and gives the user information. The camera paint detection system is also complete. The system is being used on another robot in conjunction with another project.

XVIII. Marketability

Regarding the marketability of the robot, there are a number of items that could be improved. The most important is the aesthetic quality of the robot. Putting the hardware and wiring harnesses under a shell out of sight is the next step towards making the robot market ready. Our software algorithm for the movement of the robot is designed from an even number of parking spaces only. If we were to take the design further, we would want to add the possibility of painting an odd number of parking stalls. The paint system has a limited amount of paint due to the use of aerosol cans. The real estate on the robot lends itself well to holding a case of replacement striper cans, but ideally incorporating the reservoir and sprayer system would be best for the various job types. Further perks would be a more robust user interface. Possibly a touch screen GUI with additional features, such as a paint level indicator and battery charge indicator.

XIX. Conclusion

As mentioned earlier, the potential for this technology for additional features and expanded applications is vast. This year, and the features achieved herein, mark what was able to be accomplished by a team of five students who were learning about the project as they progressed. Not all of the decisions made in the design of this project were efficient ones. There were some mistakes made that cost a great deal of time to remedy. Although this prevented some of the extra features from being implemented within our time frame, the information we gathered on our journey has all been documented here. As we lay our efforts to rest at the end of this year, we have also laid the foundation for others to use this document to inform similar projects in the future. This document will serve as an invaluable guide to the further development of this idea.

XX. References

[1] Sozo Sports Complex. Youtube: Turf Tank, 2017

[2] Interview Documentation. Attach interview Q & A responses

[3] Coahulla Creek High School. Youtube: Turf Tank, 2017

[4] Bls.gov. (2017). Painters, Construction and Maintenance. [online] Available at: https://www.bls.gov/oes/current/oes472141.htm [Accessed 7 Sep. 2017].

[5] United-paving.com. (2017). Asphalt, Seal Coat, Striping Orange County | United Paving Co.. [online] Available at: https://www.united-paving.com/recent-projects/jg-construction-2016-asphalt-seal-coat-striping [Accessed 2 Sep. 2017].

[6] Yeow, J., NG, P., Tan, K., Chin, T. and Lim, W. (2017). Effects of Stress, Repetition, Fatigue and Work Environment on Human Error in Manufacturing Industries. [online] Scialert.net. Available at: http://scialert.net/fulltext/?doi=jas.2014.3464.3471&org=11 [Accessed 10 Sep. 2017].

[7] Serensits, T. (2017). From the Field: Paint like a pro | Youth Football | USA Football | Football's National Governing Body. [online] Web.usafootball.com. Available at: https://web.usafootball.com/blogs/americas-game/post/6935health-safety/field-paint-pro [Accessed 10 Sep. 2017].

[8] Ieeexplore.ieee.org. (2017). Assisted painting of 3D structures using shared control with a hand-held robot - IEEE Conference Publication. [online] Available at: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7989566 [Accessed 9 Sep. 2017].

[9] Guo, C. (2017). Research on the improvement of human error in the process of production operation — Taking the fatigue operation in the production as the object of study - IEEE Conference Publication. [online] Ieeexplore.ieee.org. Available at: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7854481 [Accessed 10 Sep. 2017].

[10] R. Liles, “What's the Best Way to Lay Out a Parking Lot?,” For Construction Pros, 01-Aug-2017. [Online]. Available: https://www.forconstructionpros.com/pavement-maintenance/article/20867977/why-to-chalk-a-parking-lot-layout.[Accessed: 23 Jan. 2018].

[11] Brahney, S. (2015). What's The Average Cost To Line Stripe A Parking Lot?. [online] Fixasphalt.com. Available at: http://www.fixasphalt.com/blog/cost-to-line-stripe-a-parking-lot [Accessed 28 Jan. 2018].

[12] Suttle, R. (2017). Professional Stadium Groundskeeper Pay. [online] Work.chron.com. Available at: http://work.chron.com/professional-stadium-groundskeeper-pay-23474.html [Accessed 9 Sep. 2017].

[13] White, P. (2014). Line of Attack: Strategies for Painting Perfect Fields. SportsField Management. [online] 9(7), pp.10-13. Available at: http://eds.a.ebscohost.com.proxy.lib.csus.edu/ehost/detail/detail?vid=0&sid=7300b205-1416-4190-bcdf-97dd6b9c6c73%40sessionmgr4007&bdata=#AN=110495013&db=s3h [Accessed 9 Sep. 2017].

[14] Whitehead, B. (2005). Maintaining School Athletic Fields on Limited Budgets. SportsTURF, [online] 30(9), pp.26-27. Available at: White, P. (2014). Line of Attack: Strategies for Painting Perfect Fields. SportsField Management, [online] 9(7), pp.10-13. Available at: http://eds.a.ebscohost.com.proxy.lib.csus.edu/ehost/detail/detail?vid=0&sid=7300b205-1416-4190-bcdf-97dd6b9c6c73%40sessionmgr4007&bdata=#AN=110495013&db=s3h [Accessed 9 Sep. 2017].

[15] Hinze, J., Olbina, S., Orozco, J. and Beaumont, K. (2017). Earthmoving Equipment Fatalities in the Construction Industry | Practice Periodical on Structural Design and Construction | Vol 22, No 4. [online] Ascelibrary.org. Available at: http://ascelibrary.org/doi/abs/10.1061/%28ASCE%29SC.1943-5576.0000336 [Accessed 2 Sep. 2017].

[16] Mills, T., Turner, M. and Pettinger, C. (2017). Advancing predictive indicators to prevent construction accidents. [online] Available at: https://www.researchgate.net/profile/Michelle_Turner2/publication/317955292_Advancing_predictive_indicators_to_prevent_construction_accidents/links/5954482b0f7e9b2da1b04992/Advancing-predictive-indicators-to-prevent-construction-accidents.pdf [Accessed 9 Sep. 2017].

[17] Bernhard, M., Evdorides, H., Roffe, J. and Showmaker, J. (2006). A cost effective means of optimizing road safety. [online] www.irfnet.org. Available at: https://www.irfnet.ch/files-upload/pdf-files/PTCRS_publication.pdf [Accessed 5 Sep. 2017].

[18] “Summary.” U.S. Bureau of Labor Statistics, U.S. Bureau of Labor Statistics, 14 Nov. 2017 www.bls.gov/ooh/construction-and-extraction/painters-construction-and-maintenance.htm#tab-9

[19] "Turf Tank", Turf Tank, 2017. [Online]. Available: https://turftank.com/. [Accessed: 17- Sep- 2017].

[20] S. Smith, "Fieldroid - Autonomous Field Painting Robot", Contrib.andrew.cmu.edu, 2017. [Online]. Available: http://www.contrib.andrew.cmu.edu/~smsmith/fieldroid/index.html#. [Accessed: 17- Sep- 2017].

[21] "Line it up robot", Line it up, 2017. [Online]. Available: https://mapps2014.wordpress.com/. [Accessed: 17- Sep- 2017].

[22] "The Best Drones of 2018", PCMAG, 2018. [Online]. Available: https://www.pcmag.com/roundup/337251/the-best-drones. [Accessed: 28- Jan- 2018].

[23] "Case of 18oz UMA Tip Aerosol Stencil Paint For Asphalt or Concrete", Asphaltsealcoatingdirect.com, 2018. [Online]. Available: https://www.asphaltsealcoatingdirect.com/products/case-18oz-uma-tip-aerosol-stencil-paint-asphalt-or-concrete. [Accessed: 28- Jan- 2018].

[24] "Marking Machine Control Unit - MCU-200BN from Jeil Mtech Co. Ltd.", Ipfonline.com, 2018. [Online]. Available: http://www.ipfonline.com/products/index/marking_machine_control_unit_1. [Accessed: 28- Jan- 2018].

[25] "Overview | Adafruit BNO055 Absolute Orientation Sensor | Adafruit Learning System", Learn.adafruit.com, 2018. [Online]. Available: https://learn.adafruit.com/adafruit-bno055-absolute-orientation-sensor/overview. [Accessed: 28- Jan- 2018].

[26] "Sabertooth Dual 25A 6V-24V Regenerative Motor Driver", Robotshop.com, 2018. [Online]. Available: https://www.robotshop.com/en/dimension-engineering-sabertooth-2x25.html. [Accessed: 28- Jan- 2018].

[27] "Parallax 2 x 16 Serial LCD (Backlit) | 27977 | Parallax Inc", Parallax.com, 2017. [Online]. Available: https://www.parallax.com/product/27977. [Accessed: 10- Oct- 2017].

[28] ["Motor Mount & Wheel Kit - Aluminum | 28962 | Parallax Inc", Parallax.com, 2017. [Online]. Available: https://www.parallax.com/product/28962. [Accessed: 10- Oct- 2017].

[29] Learn.adafruit.com. (2018). Overview | Adafruit BNO055 Absolute Orientation Sensor | Adafruit Learning System. [online] Available at: https://learn.adafruit.com/adafruit-bno055-absolute-orientation-sensor/overview [Accessed 29 Apr. 2018].

[30] Amazon.com. (2018). [online] Available at: https://www.amazon.com/ExpertPower-EXP12120-Volt-Rechargeable-battery/dp/B010NRZW02/ref=sr_1_1?s=home-garden&ie=UTF8&qid=1524965901&sr=8-1&keywords=expertpower%2B12v%2Bbattery&th=1 [Accessed 29 Apr. 2018].

[31] “IMU Errors and Their Effects”, Novatel, 2014. [Online]. https://www.novatel.com/assets/Documents/Bulletins/APN064.pdf [Accessed: 21- Oct- 2017].

[32] S. Haberberger, “An IMU-based spacecraft navigation architecture using a robust multi-sensor fault detection scheme”, M.S., Missouri University of Science and Technology, 2016 [Accessed: 21- Oct- 2017].

[33] “Parallax LSM9DS1 9-axis IMU Module (#28065)”. Parallax, 2017 pp. 1-11 [Online]. Available: https://www.parallax.com/sites/default/files/downloads/28065-LSM9DS1-9-axis-IMU-Guide-v1.0.pdf.

[34] Macdonald, S. (2017). Question Mark - Why? - GraniteGrok. [online] GraniteGrok. Available at: http://granitegrok.com/blog/2013/07/republican-vote-stealer/attachment/question-mark [Accessed 22 Oct. 2017].

[35] Up Board | Power Up Your Ideas!. (2017). Onepage. [online] Available at: http://www.up-board.org/up/ [Accessed 22 Oct. 2017].
[36] Kumar, P. (2017). [online] ecs.csus.edu. Available at: http://www.ecs.csus.edu/wcm/eee/facultyandstaff/kumar.html [Accessed 22 Oct. 2017].
[37] "36-Position Quadrature Encoder Set". Parallax, 2017 [Online]. Available: https://www.parallax.com/product/29321
[38] "Motor Mount & Wheel Kit - Aluminum". Parallax, 2017 [Online]. Available: https://www.parallax.com/product/28962
[39] "Charging Information For Lead Acid Batteries – Battery University", Batteryuniversity.com, 2017. [Online]. Available: http://batteryuniversity.com/learn/article/charging_the_lead_acid_battery. [Accessed: 22- Oct- 2017].
[40] "eLCOSH : Electrical Safety: Safety & Health for Electrical Trades (Student Manual)", Elcosh.org, 2017. [Online]. Available: http://www.elcosh.org/document/1624/891/d000543/section5.html. [Accessed: 22- Oct- 2017].
[41] "AVR Guide: Voltage Too High, Too Low - UST", UST, 2017. [Online]. Available: https://ustpower.com/comparing-automatic-voltage-regulation-technologies/need/avr-guide-voltage-high-low/. [Accessed: 22- Oct- 2017].
[42] "UP board by Intel",Credit: UP, Bridge the Gap http://www.up-board.org/up/
[43] "Question Mark", Credit: Steve MacDonald http://granitegrok.com/blog/2013/07/republican-vote-stealer/attachment/question-mark
[44] "IMU Orthogonal Linear and Angular Axes", Credit: Novatel, https://www.novatel.com/assets/Documents/Bulletins/APN064.pdf
[45] "Averaging Noise in a Sensor System", Credit: https://www.novatel.com/assets/Documents/Bulletins/APN064.pdf
[46] "Encoders by Parallax", Credit: Parallax Inc., https://www.parallax.com/product/29321
[47] "DC motors by Parallax", Credit: Parallax Inc, https://www.parallax.com/product/28962
[48] "Charging Cycle", Credit: Battery University, http://batteryuniversity.com/learn/article/charging_the_lead_acid_battery

XXI. Glossary

- Aerosol-
  A substance enclosed under pressure and able to be released as a fine spray, typically by means of a propellant gas.
- Encoders-
  An electro-mechanical device that converts the angular position or motion of a shaft or axle to an analog or digital signal.
- Graphical User Interface (GUI)-
  A user interface that includes graphical elements, such as windows, icons and buttons.
- Institute of Electrical & Electronics Engineers (IEEE)-
  A professional association with its corporate office in New York City and its operations center in Piscataway, New Jersey. It is the world's largest association of technical professionals with more than 420,000 members in over 160 countries around the world.
- Inertial Measurement Unit (IMU)-
  An electronic device that measures and reports a body's specific force, angular rate, and sometimes the magnetic field surrounding the body, using a combination of accelerometers and gyroscopes, sometimes also magnetometers.
- Linear Actuator-
  An actuator that creates motion in a straight line, in contrast to the circular motion of a conventional electric motor.
- Liquid-Crystal Display (LCD)-
  a type of screen that is used in many computers, TVs, digital cameras, tablets, and cell phones. LCDs are very thin but are actually composed of several layers. Those layers include two polarized panels, with a liquid crystal solution between them. Light is projected through the layer of liquid crystals and is colorized, which produces the visible image.
- PID Controller-
  A proportional–integral–derivative controller (PID controller or three term controller) is a control loop feedback mechanism widely used in industrial control systems and a variety of other applications requiring continuously modulated control.
- Unmanned Aerial Vehicle (UAV)-

An unmanned aerial vehicle (UAV), commonly known as a drone, is an aircraft without a human pilot aboard. UAVs are a component of an unmanned aircraft system (UAS); which include a UAV, a ground-based controller, and a system of communications between the two.

Robotic Embellishment Parking Painter Robot (REPP-Bot) User Manual

The REPP-Bot is an excellent product to perform small jobs painting small parking lots, and the interface is very simple to learn in just a few easy steps. Below you will find Figure 1 as a reference for the rest of this manual.

1. For your convenience this device has a manual mode. There are two levers located on the chassis right next to each wheel. These levers must both be pointing 90 degrees to your right and left if you are standing next to the push bar. This will enable you to push the robot to the starting position for the job.
2. Lift the safety lock on the right switch from the figure and flip the right switch forward into the "on" position to activate the computer and user interface.
3. Lift the safety lock on the left switch from the figure and flip the left switch forward into the "on" position to activate the motors. Be sure to engage the motors physically by turning the levers located on the chassis next to the wheel toward yourself.
4. After activating the right switch, the program will launch, and you will be prompted for the number of parking stalls that you wish to paint. Enter a single number on the keypad.

Figure 1: The User Interface

5. After entering the number of parking stalls you wish to paint, you will be prompted for the size of the stalls that you wish to paint. You may select number 1, 2, or 3 as your sizing options.

1: Stall length will be 16 feet.
2: Stall length will be 18 feet.
3: Stall length will be 20 feet.

6. Make sure the motors have been engaged before you select your stall sizing since they will attempt to start painting shortly after the size 1, 2, or 3 are selected. Also, do not attempt to select sizes outside of this range, since there are none.
7. The painting will begin. And you can watch the job happen.

Additional Note: There is currently not a way to examine the amount of paint in the aerosol can, so make sure you have enough paint to complete the job you select.

Test Plan & Results

I. Executive Summary

In this paper today, we'll be going over the results of the test plan we established in the second week of EEE 193B. This test plan was supposed to serve as a guide for us as we move from a laboratory prototype to a deployable prototype. Our tests were broken down over the various electronic components that we'll be using for this project, including our encoders, IMU, and our GUI. Along with our hardware tests, we'll also have a series of software tests we must execute to verify full functionality of our REPP bot.

We'll begin with the tests on our IMU. The tests involving the IMU were both hardware and software tests. The first two tests on the IMU were verifying that the sensor was providing accurate and reliable orientation angle readings. This testing was both done on a breadboard and when the IMU was physically attached to our robot. The results of this test were that our IMU proved to be a highly accurate and dependable sensor for the application of our line painting automation. Following the test on the reliability and accuracy of our robot's readings, we then moved onto testing how the software algorithm we built for applying the IMU on correcting the robot's navigational path and turning capabilities. The tests presented a critical dependency on the IMU's capability to provide accurate readings and caused us to adjust our algorithm in such a way that future failures when deploying our robot could be mitigated.

The next set of tests executed from our test plan were on the paint system itself. This set of tests was put high on our agenda because without a functioning and deploying paint unit, we wouldn't have a line painting robot. These set of tests looked at how simple it was to replace aerosol paint cans, the amount of time it took to for paint to be dispensed after activation, and test the smoothness of integrating the paint unit with the robot. The test of how simple it was to replace aerosol cans proved that the process was rather simple and the amount of time it took for paint to be dispensed allowed us to set up our code accordingly.

Parallel to the testing of our paint system, we also had a series of tests run on our GUI. The GUI tests included the verification of our membrane keypad working correctly. The test verified that it was. Following that, the next test was to double check that based on inputs received on the keypad, you could change the behavior of a system connected. The reason this was important had to do with the fact that without this capability, the GUI would be able to act as another control mechanism to our REPP bot. The results of this test concluded that our GUI will be able to be used as a control mechanism and the final test we'll be completing as we wrap up our test plan is the integration of the GUI onto our robot.

Earlier in this paper, we discussed how the IMU was a central component in our robot's navigation system. The other key component is the encoders and in this set of tests, we looked at whether our encoder sensors were providing accurate readings, how well the encoders helped the robot correct its path, if it could be used for accurate distance tracking, and how well the integration of this component would be. Upon completion of these tests, we found that the encoders passed all tests and was able to be fully integrated on our robot as another integral sensor.

The final set of tests were performed on our camera system. The camera system comprised of both a Raspberry Pi and a sainsmart camera. The tests performed included verification that openCV and the Python IDE, Thonny, was installed correctly. The other two tests were the successful interfacing of the PI with the camera, and the testing of real time line detection from the camera. The testing of real time line detection is the only test that is still in progress and will be completed by the end of this week.

The summary above outlines all the results of the tests we performed on our REPP bot and can be visually seen in our test table below.

Test	Details	Outcome
Membrane Keypad Input	Tested each key on keypad	Individual inputs match
Input Behavior Switching	Wrote a basic interface for altering robot behavior	Inputs yield desired changes in behavior. Additional error handling in progress
Integration with Robot	Integrate GUI with LCD and full robot behavior	In progress deadline March 16th
Testing individual encoders for proper readings	**Trial 1** Encoders to seem to be giving proper readings 5 seconds of testing for both encoders **Trial 2** Encoders read the close to the same readings on a time interval of 5 seconds Was off by about ± 0.07%	Successful hardware testing Readings are as expected
Test encoders for accurate distance tracking	**Trial 1** 138 inches at a total amount of ticks at (56609 + 57246)/2 **Trial 2** 137.5 inches at a total amount of ticks at (56578 +57121)/2	same amount ticks traveled with about ± 0.13 % off
Test encoders for straight path	**Trial 1** Has slight lean to the right with Kp = 2.1 and Ki = 0.123 **Trial 2**	Line Painting Robot now goes fairly straight, locally
	The slight lean has gone away	
Testing accuracy of IMU readings (stationary)	For this test, we ended up seeing how the sensor's orientation readings were. Tested on a breadboard.	The accuracy of the IMU readings proved to be fairly accurate and consistent.
Testing accuracy of IMU readings (while attached to mobile robot)	For this test, we saw how the sensor's orientation readings changed when attached to robot. Tested on REPP bot.	The accuracy of the robot seemed to be accurate over smaller distances, such as the navigation of the 6 parking stalls.
Test IMU in extreme heat environment	The test was supposed to be done by putting the IMU in extremely warm environments and seeing if the sensor readings changed at all in comparison.	We didn't end up performing this test.
Test how robot does when IMU fails	This test was to see whether the robot could perform its navigation without the IMU, in case it ever ended up putting out inaccurate readings.	Didn't end up performing test. We reoriented towards robot stopping paint job when IMU becomes defective.
Paint Refill Simplicity	The test was to determine the ease of use in	Test was successful. The system is easy

	refilling the paint when it runs out. The desire was a very simple and straightforward way for the user to add paint to the robot when it runs out.	to use in refilling the paint.
Paint Dispensing Time Test	The test was to gauge the average dispensing time of the paint system as well as the consistency of the dispensing. Ideally, the system would have about the same time dispensing paint each run, this way we can program an alert that will indicate the system needs to be refilled by the user. Likewise, the consistency of the dispensing would determine how fast the paint will run out.	This test wasn't completed. Rather, we are working on the camera system as a way to detect if the paint starts to fail before the distance tracker indicates it's reaching it's anticipated distance.
Paint Actuator Response Time	The test was crucial to the accuracy of the paint system. The paint dispensing is likely going to be controlled by a servo or stepper motor. The system needs to have a virtually instantaneous response time when triggering/terminating the paint dispensing. Anything less will mean messy lines and inaccurate measurements.	The actuator we put into the system has a rapid enough response time to uphold the accuracy of the system.
Paint Integration Test	This test was after the paint system has been integrated on the robot. The goal was to verify whether the integration has disrupted the system's performance.	The paint system does not affect the robot's other functions.
Camera and Raspberry Pi	Interfacing the Camera and Rpi	Successful
OpenCV	Installing openCV on Rpi	Successful
Real time	Line detection in real time	In progress

Table 1: Testing Progress

II. Introduction

This paper will discuss the results of the test plan that was designed in assignment two of EEE 193B. More specifically, these results are from testing each feature separately. Our robot has five distinctive features/components which are the IMU,

paint system, GUI, camera system, and encoders. Each of these features/component have their own unique set of tests to be performed in order to ensure that they are being implemented correctly.

The set of tests that were performed on the IMU were how accurately it outputs orientation angles, how it does in warm environments, software test on IMU's contribution to the control algorithm, and the fail test, which is the effect on the robot if IMU failed mid job. On the other hand, the set of tests for the GUI included the functionality of the membrane keypad, and testing the user interface software. The paint system ran tests such as the simplicity of refilling the paint when it runs out, gauging the average dispensing time of the paint system, the consistency of the dispensing, and the accuracy of the paint system. Furthermore, the tests ran on the encoders include distance tracking, speed testing, and straight path testing. Finally, the set of tests for the camera system were concerning the interfacing between the camera and microcontroller, successfully installing openCV, and detecting lines in real time.

III. Inertial Measurement Unit Test Results

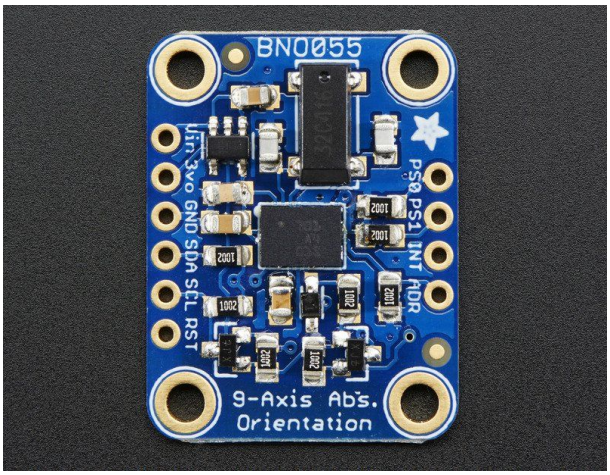


Figure 1: Adafruit IMU BN055
Credit:
https://learn.adafruit.com/adafruit-bno055-absolute-orientation-sensor/overview

In this section of our paper, we'll be discussing the results of the tests done on our IMU being utilized for this project. The first test that we began with was a simple one and that was verifying that the IMU was outputting accurate orientation angle readings that could be used for navigation. The test was split into two parts where the first was testing the accuracy of these readings on a breadboard and rotating the breadboard to see if the IMU could pick up the change in angle. The second part of this test was the accuracy of these readings while the robot was mobile. The breadboard test and attaching it to the robot allowed us to see how the IMU did in different environments, one more realistic to the job that we're accomplishing. What we found from our tests was that the orientation angle readings we were getting from our IMU could be relied upon within a 2 degree accuracy readings (plus or minus 1 degree). This made us realize that when it came to the 90 degree turns that the robot would need to pull off while painting the parking stall, we would have to find a way for the IMU to correct the robot's path prior to continuing its paint job on turns. The change in our algorithm wasn't dramatic and has been resolved already.

The next test that we followed up with on our IMU was the testing of how it did in a warm environment. We didn't end up performing this test and the reason had to do with how the robot design ended up coming out. Previously, the worry behind how it would perform in different environments had to do with the fact that the IMU, along with many of our other electronic components, was going to be encapsulated in a box on our robot. The reason behind putting all electronic components in this 3-D printed box was to clean up the aesthetics of our robot. What ended up happening was that we created a hole in this box for wires from the breadboard to connect to whatever they were controlling on our robot. This hole was created rather large and ended up serving a dual role as the gateway for all electromechanical connections running from the breadboard but also a ventilation system that would allow the robot to run too hot. This is all assuming that the robot will not be used in extreme temperature applications which, for the scope of our project in this phase, is reasonable.

Following the thermal test, a software test on our IMU's contribution to the control algorithm was to be tested. This was a rather large test so it had to be split into 2 sections as well. The first test was related to the IMU's task as a primary navigation mechanism for the line painting robot. In this test, we developed our software algorithm for the IMU to help with correction of the robot's navigational

path. We executed this test by uploading the code to our robot, making it move in a pre-programmed path, and with the help of chalk lines pre-laid down, visually verify that the algorithm is working with in the desired accuracy range. The test showed that the IMU was a very reliable sensor to work with on navigation and could maintain small path corrections over the parking stalls we'll be creating for our prototype. The accuracy was within less than 6 inches. With that being said, what we also came to discover was that as this robot kept on moving over longer and longer paths, this accuracy began to dwindle. This goes back to the fact that the our IMU can only be depended upon within plus or minus a degree of accuracy. The small error that the IMU readings has occurring causes an accumulation to occur so as you move farther and farther distances, this causes the robot to drift off of our desired. While this won't be a problem because the accumulation is very small when painting 6 parking stalls, this issue would become more obvious is we were to paint 30 parking stalls. How we're going to attempt to resolve this is by reintegrating our GPS unit back onto the robot as a global reference. While GPS's are notorious for having rather large distance resolutions, meaning the readings can only be relied upon within a couple of meters, we hope that this global reference will better serve for correction when this robot needs to move across larger distances. The other aspect to testing our software algorithm related to how well the IMU could get the robot to make 90 degree turns. This was mentioned previously in the paper but re-summarize, the IMU can only work within plus or minus a degree. This issue was found during testing and resolved. While the IMU does not initially make the 90 degree turn perfectly, the robot knows to correct itself prior to continuing its paint job on every turn.

The final test we mentioned was a fail test where we see what would happen to the robot if the IMU was to end up failing mid-job. This test was never done for a variety of reasons. The first being that if the IMU was to end up failing, the robots other sensors don't work within the degree of accuracy required for completion of a line painting job done by us. What we've decided to do there instead is to end up sending a notification to the user via the GUI created that a sensor has become defective. Thinking through this test did provide us with insight into how critical the IMU and has made us consider taking further precautions outside of the GUI notification to assure no line painting jobs are done incorrectly.

IV. Paint System Test Results

The tests related to the paint system are two-fold. The first three tests can be determined before the paint system is incorporated into the full robot design.

Figure 2: Striping Paint
Credit: https://images-na.ssl-images-amazon.com/images/I/51-aFOCAY9L._AC_UL320_SR272,320_.jpg

The first test was to determine the ease of use in refilling the paint when it runs out. The desire was a very simple and straightforward way for the user to add paint to the robot when it runs out. Observing a number of people with little instruction attempting to add paint to our paint system will determine the simplicity. The refill system proved to be very self explanatory. And the ease of replacing the aerosol spray can was as desired.

The second test was to gauge the average dispensing time of the paint system as well as the consistency of the dispensing. Ideally, the system would have about the same time dispensing paint each run, this way we can program an alert that will indicate the system needs to be refilled by the user.

Likewise, the consistency of the dispensing would determine how fast the paint will run out. Given that the distance covered for each can is set, we were going to track the distance and make an alert when the system covered the distance. To verify that the paint system doesn't fail, we want to incorporate a camera system to tell if the paint fails.

The third test was crucial to the accuracy of the paint system. The paint dispensing is likely going to be controlled by a servo or stepper motor. The system needs to have a virtually instantaneous response time when triggering/terminating the paint dispensing. Anything less will mean messy lines and inaccurate measurements. The results from the tests were satisfactory. The actuator that we incorporated has an extremely fast response time, so the spraying commands are very quick.

The final test was after the paint system has been integrated on the robot. The goal was to verify whether the integration has disrupted the system's performance. Putting the robot through a number of painting trials would allow us to see if the paint system has lost any of its functionality or accuracy after being put on the robot. So far, the paint system has not inhibited the functionality of the robot's other processes. We have to integrate the main movement with the painting commands as one of the final steps. We hope to do this by the end of the coming week.

V. Camera Test Results

Our original test plan created for the camera system did not go quite as smoothly as we expected it to. The Intel Edison was not properly communicating with the Macbook Pro, and this is mainly due to the fact that the Intel Edison is discontinued. The last time the drivers required to be installed on a Mac, prior to establishing a connection with the Edison, were updated was 2016. Since the Macbook we are working with is fairly new, it was unable to detect the drivers. Luckily, we had the Raspberry Pi micro-controller, and the SainSmart Camera as a back up.

The set of tests that were designed for the Intel Edison and RealSense Camera all went smoothly. The first test was interfacing the camera and microcontroller. This was verified when the images captured with the camera were accessible from the Raspberry Pi.

The second test was to verify that OpenCV and the IDE we used to program in python are successfully installed on the microcontroller. OpenCV is a library of built in functions created for real time machine vision, and machine learning applications. Python is a programming language known for machine vision applications. OpenCV was installed following a guide discovered online[]. To verify that openCV, and the IDE was successfully installed, a simple color detection program was written in Python. The program was written to detect the color red in an image, and that is what it did.

The last test was to verify that the camera is detecting the lines in real time. The line detection algorithm that we currently have takes a raw image and detects the lines in the image. However, there is a lag when the output is displayed to the user. This issue can be resolved by using a faster microprocessor, but due to our time constraint, it is something that we will not explore this semester.

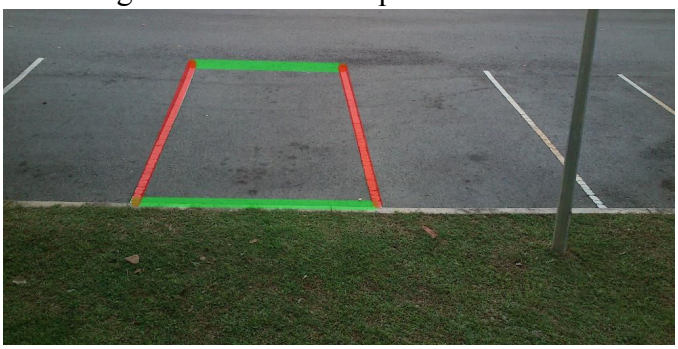


Figure 3: Line detection OpenCV
Credit: https://i.stack.imgur.com/Cty4T.jpg

VI. Graphical User Interface Test Results

The graphical user interface (GUI) is a critical component for the users to have a positive experience operating our device. Without a robust GUI for the user to communicate exactly what they want our robot to do, the user will only have the flexibility to make the robot perform a single pre-programmed task. While this is already an upward move from our original problem statement, this is unacceptable for our device. We want our users to be able to change the number of parking

stalls they want to paint manually via a number pad and display screen. At the power-up of the device, a display screen will ask the user to input the desired number of parking spaces. The user will then input this number, and the robot will proceed from that spot to paint the desired number of parking spaces.

When we first started to develop this feature, needed to first develop a plan to test the individual input devices to ensure that they read appropriately from user input. To do this, we wrote a simple program connected with the user input device to display the keys that are pressed. The test ran through all available user input keys and verified that each is registering accurately for the key that is pressed.

Figure 4: Membrane Keypad
Credit:
https://www.tindie.com/products/mmm999/12-key-array-membrane-keypad/

This test was done with a membrane keypad device, which is pictured above. The results of this test were successful. Each button pushed yielded the correct result printed to the screen ensuring that our membrane keypad is functioning correctly.

The second test followed suit and tests a user-interface software that makes decisions based on the users input. The designed software mimics the robot's behavior in the abstract. The goal of this test was to determine that the software behaves in a desired manner and that user errors are accounted for and appropriately handled. To aid with this testing phase, in addition to testing the software itself, we had willing non-engineers attempt to interact with the software and noted their feedback for any issues they had. These issues will be addressed in the next iteration of the software. These tests were completed by March 9th with the necessary changes made to the software. The keypad had no trouble making decisions based on user input, but the users noted that a larger LCD display would be better. We are currently looking at ways to improve upon this design including upping the size of our current LCD and developing a touchscreen display. The touchscreen display is a little ambitious, because a very complex library is needed to program a full-scale GUI, but we are committed to thoroughly evaluating this option.

The final phase of GUI testing stems naturally from the previous two by integrating the abstract software into actual decisions and instructions for the robot to follow. These tests should be thorough including testing for error handling and a variety of user inputs. The software should be robust enough allow the user great freedom in choosing how many parking stalls to paint. These tests should be completed by March 16th to be completed with enough time for troubleshooting. With all of these tests of the GUI from individual components to complete system, we should be able to verify that the user interface that we produce is a valuable addition to the feature set. We may need some additional time over the spring break to get these tests fine-tuned, but the status of the GUI as of now is ready to integrate with the robot code provided the tests with paint and navigation are complete.

VII. Encoders Test Results

The test plan for encoders consists of distance testing, speed testing, straight path testing, and tracking turns. The test should be able to validate that the robot will behave as it is suppose to. When the user demands a specific speed from the user interface, the encoders will be the main component to keep track of the speed. This is the reason why speed testing and distance testing is so crucial. After the test for track speed and distance has been completed, the robot should then be able to travel a certain distance demanded by the user while, at the same time, keeping track of speed accurately. The

procedure to make sure the robot has met these requirements will be to draw a line of a certain length. We will then compare these results to the users input of specific size of the parking lot and speed at which the user wanted the robot to be working at.

The testing results as of March 11, 2018 for the encoders have been completed with some changes to the plans. The testing plans now consist of testing the encoders itself for proper readings, testing encoders for distance tracking, and testing encoders for straight path The reason for taking out the speed tracking and turn tracking of the test plan was because the speed was part of the distance tracking and our line painting robot will only be going at a fixed voltage which means a fixed speed. The turn tacking will be done by the IMU. The results of the test will now be discussed. The testing of the encoders consist of two trials. The first trial of the test was a short 5 second burst to make sure that the encoders gave correct readings. The way the test was carried out was by making sure if right encoder had more ticks than left encoder. The robot will be varying left which was the case for our robot. After the completion of the first trail I implemented an PID algorithm to make the robot go fairly straight both encoders were about plus or minus 0.07 percent from one another.

The second test was the testing for distance tracking. This test also consist of two trials where I made sure that the line painting robot was able to keep track of its distance. The test was carried out by a means of making sure line painting robot went straight and marking the initial spot then letting it go for about 10 seconds. After 10 seconds I then marked the final position in which it stopped and measured from initial to final position. I repeated the test twice which allowed me to closing zero into the accuracy of how much the robot has traveled. With both trials the outcome of the distance tracking came out to be about half a inch off each time I tested after both trials. This in turn will now allow our line painting robot to paint lines of certain lengths. The last test to be discussed is the straight path with the encoders. The encoders allowed the robot to go fairly straight. This was due to distance tracking because it allowed for our line painting robot to keep track of amount of ticks per 50 millisecond. The straight path was then tested after writing the algorithm to keep track of ticks per second. The first trial had a slight lean to the right with Kp = 2.1 and Ki = 0.123. The second test now consisted of making the line painting robot to go straight without know its global reference. The second trial the line painting robot now no longer has a slight lean to the right. This concludes the test plans for just the encoders alone.

The next portion of testing for encoders will be cared out when we integrate the encoders with the other sensors on line painting robot.

VIII. Fully Functional Robot Test Results

The final phase of tests for the robot is after all the systems are integrated successfully and deal with the robots performance.

The first test is to confirm the robot’s movement on various surfaces. It will be necessary to put the robot through it's navigation paces on the various surfaces that we anticipate it working. In the unlikely case that it would experience any issues, they would need to be addressed as quickly as possible. This test will have to be performed by the 5th of March.

In order to correct any issues concerning navigation, we will need to conduct further tests. The other test will be to observe the robot when disturbance is introduced under the robot. Putting objects under the tires, or running the robot over the an uneven surface will introduce enough disturbance to give us an idea of how it will respond. The results will allow us to determine any changes needed for the design. This test will be concluded by March 5th as well

One of the features of the robot is object detection and avoidance. To test the responsiveness and accuracy of our sensors, we will run the robot near various kinds of obstacles. The results will help us fine tune the placement and software regarding the system. These test runs will need to be completed by March 9th at the latest.

The final test is regarding the power system of the robot. We want to make sure the robot can perform the painting job in the course of a single battery charge. The results will indicate whether we need to change any components or add batteries to

the system. This test will be completed by March 9th.

IX. Conclusion

All of our initial parts tests seem to have shown promise when tested as individual components. The challenge ahead is to integrate all of these individual tests effectively into the overall design of the robot. To do this, we must maintain a strict system of deadlines to prevent one incomplete part from straining other parts for time. Any minor errors we may have had in individual testing have a higher probability of compounding when integrated with other parts. So this next phase of testing must also be very methodical, so that we can isolate issues easily when they do occur. And lastly, we can continue to look for additional ways to meet and push even further the requirements of our feature set and continue to collaborate to find unique solutions to any issues in need of addressing.

X. References

[1] Townshend, K. (2018). *Overview | Adafruit BNO055 Absolute Orientation Sensor | Adafruit Learning System*. [online] Learn.adafruit.com. Available at: https://learn.adafruit.com/adafruit-bno055-absolute-orientation-sensor/overview [Accessed 4 Feb. 2018].

[2] Images-na.ssl-images-amazon.com. (2018). *Striping Paint Image*. [online] Available at: https://images-na.ssl-images-amazon.com/images/I/51-aFOCAY9L._AC_UL320_SR272,320_.jpg [Accessed 4 Feb. 2018].

[3] Line, O. (2018). *OpenCV C++ Draw Rectangle Based on two line*. [online] Stackoverflow.com. Available at: https://stackLoverflow.com/questions/18036833/opencv-c-draw-rectangle-based-on-two-line [Accessed 4 Feb. 2018].

[4] Tindie. (2018). *12 Key Array Membrane Keypad by Mmm999 on Tindie*. [online] Available at: https://www.tindie.com/products/mmm999/12-key-array-membrane-keypad/ [Accessed 4 Feb. 2018].

Features:

※ Multi-column structure for option, 38, 52, 62, solid shaft, hollow shaft, Semi-Hollow shaft

※ Multi-output mode: Voltage, Open-collector, Push-full, Driver output

※ Small volume, light weight, flexible connecting shaft, easy installation

※ Metal housing, high anti-interference

※ Pulse range: Form 10 to 5000pps. Widely applied to light industry machinery, textile, wire and length measuring.

1. Working principle & Description

Photoelectric rotary encoders are high precision devices which are consist of light, mechinery, electricity. Working by photo and electricity rotation, can convert the output shaft of the angular displacement, angular velocity and other mechanical quantity to electrical pulses which display with digital output. Generally Photoelectric rotary encoders have incremental rotary encoder and absolute encoder. According to design requirements, in this description is only to introduce incremental encoder. When the Incremental Encoder make rotation, there is a corresponding pulse output, to identify the direction of its rotation and pulse increase or decrease through the external direction-judgable circuit and counters to achieved. The counting point can be set freely and can be multi-circle accumulated and measuring, but also can be issued the Z signal produced by per revolution as a reference for machinery zero point.

When the rotary encoder shaft turning a circle it will produce a fixed amount of pulses. For the photoelectric rotary encoders, the amount of output pulses is the same with the amount of rotating grating engrave lines. If need to increase the resolution, we can use 90-degree phase-shift of A, B two signals to frequency doubling the original signal or use higher resolution encoder.

Output pulse calculation

Output pulses for per revolution (cpr) = (encoder corresponding displacement produced by per revolution: the length or angle)/ set resolution

Photoelectric rotary encoder is widely used in automatic control, automatic measurement and other automation fields, such as CNC machine tools, servo motors, high-speed elevators robotics, steel rolling machinery, textile machinery, printing machinery, light industry machinery, automotive, financial electronic, oiling machinery, flow machine, test machine and office automation instrumentation industry.

2. Ordering Code

IM □□-□□□□

Additional Function	Default	Standard cable output
	K	Aviation plug output
	B	Semi-Hollow shaft coupling
Numbers of pulses	600	180、200、300、360、400、500、600、800、900、1000、1024、1200、1500、1800、2000、2400、3000、3600、5000
Supply voltage	A	5V DC±5%
	B	12～24V DC ±5%
Output circuit	R	Voltage output
	K	Open Collector NPN output
	S	Push-pull output
	D	Line driver output
Diameter	40	Diameter ϕ38mm, shaft diameter ϕ 6mm (Axis hole ϕ 8mm)
	50	Diameter ϕ 50mm, shaft diameter ϕ 8mm (Axis hole ϕ 8mm)
	60	Diameter ϕ 66mm, shaft diameter ϕ 8mm
Series	S	S Solide shaft Series Rotary Encoder
	H	S Hollow shaft Series Rotary Encoder

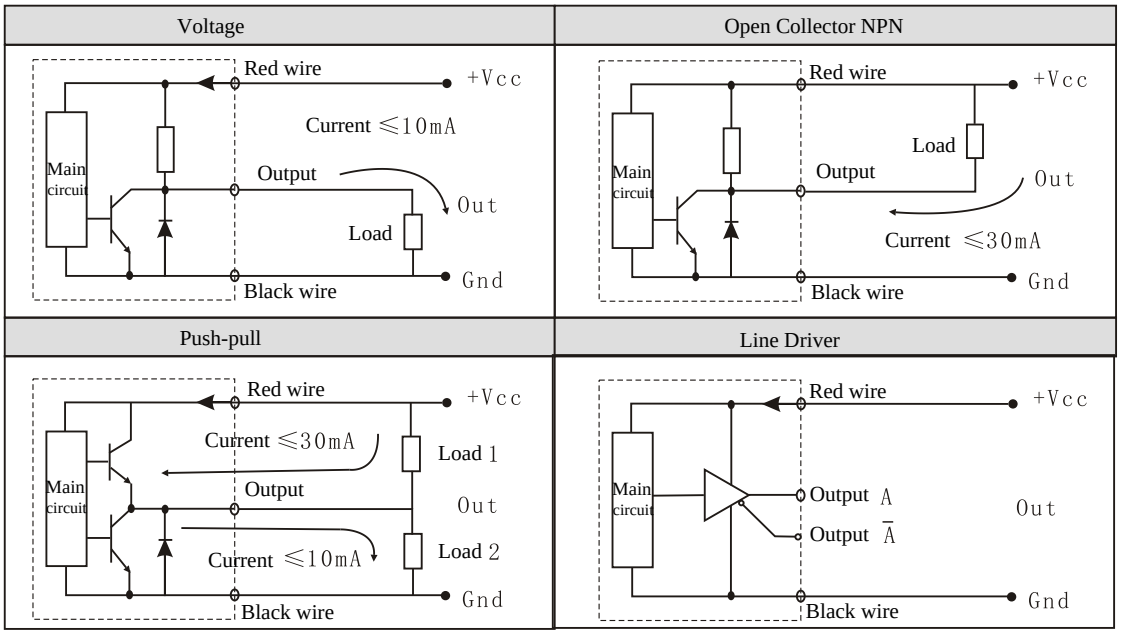
Example:S40-RB600, means that the product is Solide shaft series rotary encoder, and the Diameter is : Diameter ϕ38mm, shaft diameter ϕ 6mm (Axle hole ϕ 8mm), Voltage output circuit, the number of pulses is 600 P/R.

3. Terminal assignment

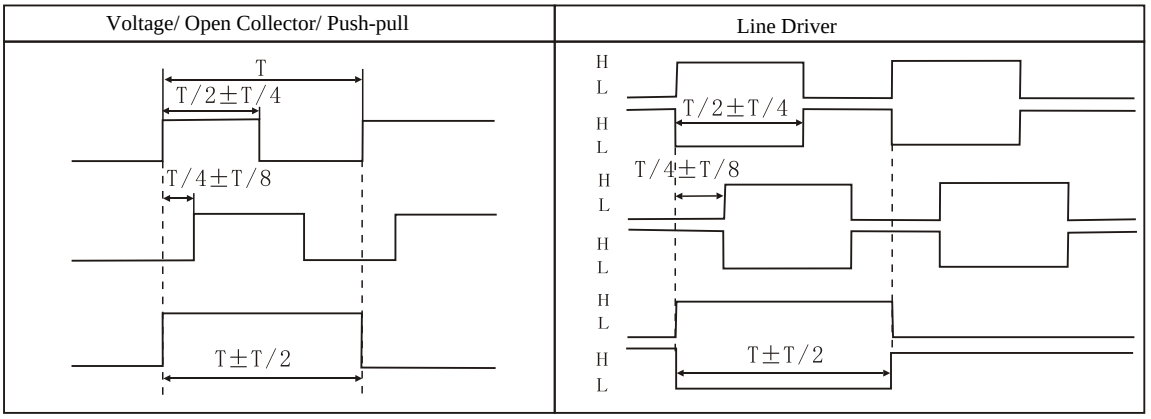
Signal	Power+	Power-	SIGA	SIGB	SIGZ	Shield
Cable color	Red	Black	Green	White	Yellow	Copper mesh
7 pin plug	1	4	3	5	2	6
9 pin plug	1	4	5	3	2	9

Non-A	Non-B	Non-Z
Brown	Grey	Orange
7	6	8

4. Output circuit



5. Output Waveform



6. Caution

It may cause malfunction if below instructions are not followed.

1. Photoelectric Rotary Encoders are high precision devices. Therefore please treat this product carefully. Do not put strong. It is prohibited to knock or hit or hammer. Inappropriate or wrong installation can influence the capability and operating life of the rotary encoder.
2. Encoder Solid axes and the user axis should be avoided rigid connections, please use elasticjunction panel or flexible coupling to avoid user axis jumping or bounding. Otherwise the encoder axes and the encoding board damage may result.
3. Hollow shaft and electrical machinery should be mounted clearance fit, can not be too tight or too loose, also the locating key cannot be too tight. Beating to install is strictly prohibited.
4. Make sure that the difference between encoder axis and users axes must less than 0.02mm, the angle of both axes must less than 1.5°.
5. Make sure do not exceed the limited rotate speed specified, if exceed the specified rotate speed, the electrical signals might lose. The limited rotate speed under normal operation of rotary encoder is:
   Nmax=(60F× 102/L)× r/min (F is the frequency response ,L is the reticle number of raster)
6. Be sure if the connection and wiring is correct, wrong connection may cause damage to the internal circuit of the rotary encoder. Please connect the wires according to the diagram given out in the product.

Features:
1. Housing diameter 38mm, to be use in light industry;
2. Coupling mode: semi-hollow shaft
3. Direct cable output or different kinds of socket to be optional connection
4. Multi-output modes for option, more flexibility.
5. Oput terminal with water proof protedction, more safety.

1. Ordering Code

IM H40 -□ □ □ □

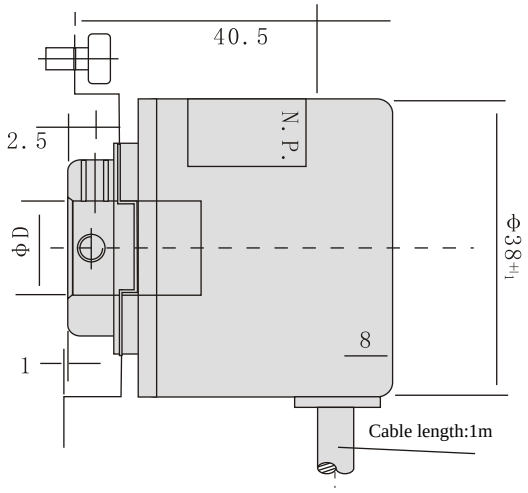
Additional Function	Default	Standard cable output
	B	Semi-Hollow shaft coupling
Numbers of pulses	600	60、100、120、180、200、300、360、400、500、600、720、800、900、960、1000、1024、1200、1500、1800、2000、2400、2500
Supply voltage	A	5V DC ±5%
	B	12～24V DC ±5%
Output circuit	R	Voltage output
	K	Open Collector NPN output
	S	Push-pull output
	D	Line driver output
Diameter	40	Diameter ϕ38mm
Series	H	H Series hollow-shaft rotary encoder

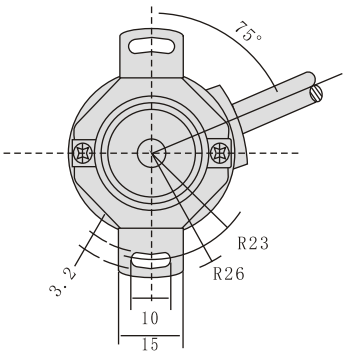
Example:IM H40-RB1024, means that the product is IM H Hollow shaft series rotary encoder, and the housing diameter is 38mm, Voltage output circuit, 12V or 24 V power supply; the number of pulses is 1024P/R.

2. Technical Parameters

Power supply	5V DC ±5%、12～24V DC±5%	Max. rotating speed	6000rpm
Output voltage	Vh ≥85%Vcc Vl≤0.3V	Vibration resistance	50m/s^2, 10-200Hz, 2 times each in X,Y,Z directions
Current Consumption	≤150 mA	Shock resistance	980m/s^2, 6ms, 2 times each in X,Y,Z directions
Response frequency	0～100K Hz	IP rating	IP54, dustproof
Output wave	Square wave	Operating life	MTBF ≥ 10000h
Duty ratio	0.5T±.1T	Working temperature	-10～70℃
Starting torque	5×10^{-3}N.m	Storage temperature	-30～85℃
Loading radial	Radial ≤20N	Ambient humidity	30-85% (with no condensation)
Loading axial	Axial ≤10N	Weight (appr)	180g

3. Appearance & Dimension





Features:
1. Housing diameter 58mm, to be use in light industry;
2. Coupling mode: semi-hollow shaft
3. Direct cable output or different kinds of socket to be optional connection
4. Multi-output modes for option, more flexibility.
5. Oput terminal with water proof protedction, more safety.

1. Ordering Code

IM H 60-□ □ □ □

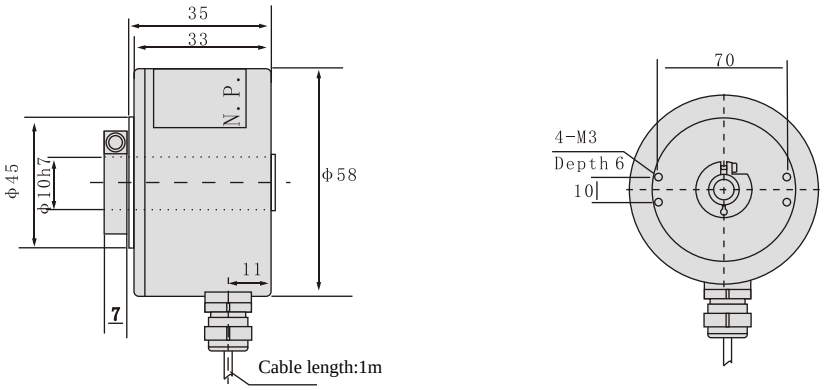
Additional function	Default	Standard cable output
Numbers of pulses	600	60、100、120、180、200、300、360、400、500、600、720、800、900、960、1000、1024、1200、1500、1800、2000、2400、2500
Supply voltage	A	5V DC ±5%
	B	12～24V DC ±5%
Output circuit	R	Voltage output
	K	Open Collector NPN output
	S	Push-pull output
	D	Line driver output
Diameter	60	Diameter ϕ58 mm
Series	H	H Series hollow-shaft rotary encoder

Example:IM H60-RB1024, means that the product is IM H Hollow shaft series rotary encoder, and the housing diameter is 58mm, Voltage output circuit, 12V or 24V power supply; the number of pulses is 1024P/R.

2. Technical Parameters

Power supply	5V DC ±5%、12～24V DC±5%	Max. rotating speed	6000rpm
Output voltage	Vh≥85%Vcc、 Vl≤0.3V	Vibration resistance	50m/s^2, 10-200Hz, 2 times each in X,Y,Z directions
Current Consumption	≤150 mA	Shock resistance	980m/s^2, 6ms, 2 times each in X,Y,Z directions
Response frequency	0～100K Hz	IP rating	IP54, dustproof
Output wave	Square wave	Operating life	MTBF≥10000h
Duty ratio	0.5T±.1T	Working temperature	-10～70℃
Starting torque	4×10^{-3}N.m	Storage temperature	-30～85℃
Rotor moment of inertia	Appr. 7.5×10^{-6}Kgm^2	Ambient humidity	30-85%RH (with no condensation)
Max. load	Radial ≤40N、 Axial≤25N	Weight (appr)	250g

3. Appearance & Dimension



Features:
1. Diameter 58mm, shaft diameter 6mm. To be use in light industry;
2. Small volume, light weight;
3. D-shaped incision, easy installation;
4. Multi-output modes for option, more flexibility;
5. Output cable Side Entry

1. Ordering Code

IM S 40 −□ □ □ □

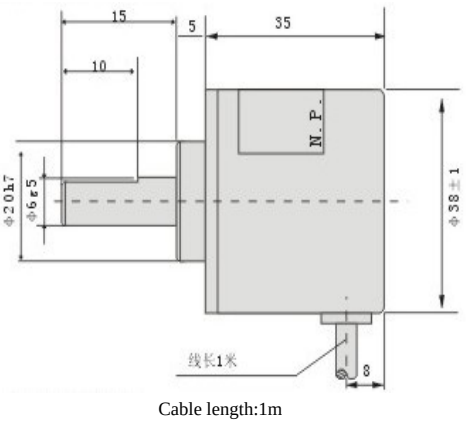
Additional function	Default	Standard cable output
Numbers of pulses	600	60、100、120、180、200、300、360、400、500、600、720、800、900、960、1000、1024、1200、1500、1800、2000、2400、2500
Supply voltage	A	5V DC ±5%
	B	12～24V DC ±5%
Output circuit	R	Voltage output
	K	Open Collector NPN output
	S	Push-pull output
	D	Line driver output
Diameter	40	Diameter 38mm, shaft diameter 6mm
Series	IM S	ES Series Solid-shaft rotary encoder

Example:IM S40-RB600, means that the product is IM S Solid shaft series rotary encoder, and the housing diameter is 38mm, shaft diameter 6mm; Voltage output circuit, 12V or 24V power supply; the number of pulses is 600P/R.

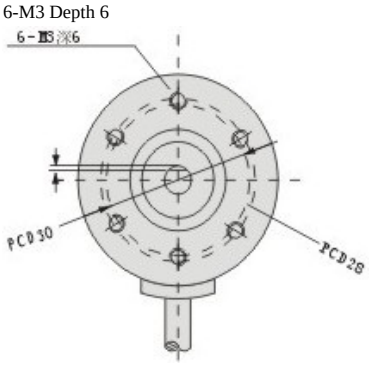
2. Technical Parameters

Power supply	5V DC ±5%、12～24V DC±5%	Max. rotating speed	6000rpm
Output voltage	Vh≥85%Vcc、 Vl≤0.3V	Vibration resistance	50m/s², 10-200Hz, 2 times each in X,Y,Z directions
Current Consumption	≤120mA	Shock resistance	980m/s², 6ms, 2 times each in X,Y,Z directions
Response frequency	0～100K Hz	IP rating	IP54, dustproof, waterproof, oilproof
Output wave	Square wave	Operating life	MTBF≥10000h
Duty ratio	0.5T±.1T	Working temperature	-10～70℃
Starting torque	1.5×10^{-3} N.m	Storage temperature	-30～85℃
Rotor moment of inertia	Appr. 3.5×10^{-6}Kgm²	Ambient humidity	30-85%RH (with no condensation)
Max. load	Radial ≤ 20N Axial ≤ 10N	Weight (appr)	100g

3. Appearance & Dimension



Cable length:1m



Features:
1. Diameter 50mm, shaft diameter 8mm. To be use in light industry;
2. Small volume, light weight;
3. With cable output;
4. Multi-output modes for option, more flexibility;
5. Max. ratating speed up to 6000rpm

1. Ordering Code

IM S 50 -□ □ □ □

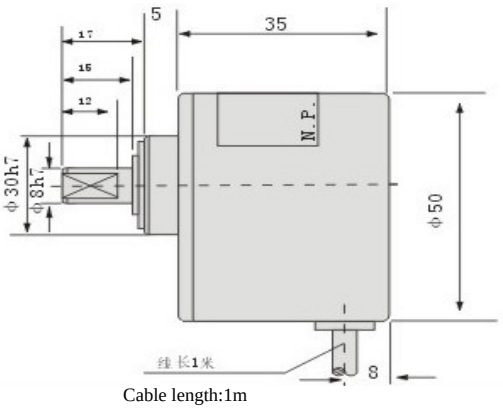
Additional function	Default	Standard cable output
Numbers of pulses	600	60、100、120、180、200、300、360、400、500、600、720、800、900、960、1000、1024、1200、1500、1800、2000、2400、2500
Supply voltage	A	5V DC ±5%
	B	12～24V DC ±5%
Output circuit	R	Voltage output
	K	Open Collector NPN output
	S	Push-pull output
	D	Line driver output
Diameter	50	Diameter 50mm, shaft diameter 8mm
Series	IM S	IM S Series Solid-shaft rotary encoder

Example: IM S50-RB360, means that the product is IM S Solid shaft series rotary encoder, and the housing diameter is 50mm, shaft diameter 8mm; Voltage output circuit, 12V or 24V power supply; the number of pulses is 360P/R.

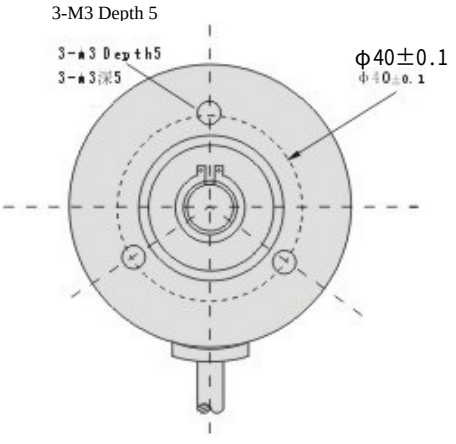
2. Technical Parameters

Power supply	5V DC ±5%、12～24V DC±5%	Max. rotating speed	6000rpm
Output voltage	Vh≥85%Vcc、 Vl≤0.3V	Vibration resistance	50m/s², 10-200Hz, 2 times each in X,Y,Z directions
Current Consumption	≤180mA	Shock resistance	980m/s², 6ms, 2 times each in X,Y,Z directions
Response frequency	0～100K Hz	IP rating	IP54, dustproof, waterproof, oilproof
Output wave	Square wave	Operating life	MTBF≥10000h
Duty ratio	0.5T±.1T	Working temperature	-10～70℃
Starting torque	5×10^{-3} N.m	Storage temperature	-30～85℃
Rotor moment of inertia	Appr. 6×10^{-6}Kgm²	Ambient humidity	30-85%RH (with no condensation)
Max. load	Radial≤35N Axial≤25N	Weight (appr)	100g

3. Appearance & Dimension



Cable length:1m



Web Site: www.parallax.com
Forums: forums.parallax.com
Sales: sales@parallax.com
Technical: support@parallax.com

Office: (916) 624-8333
Fax: (916) 624-8003
Sales: (888) 512-1024
Tech Support: (888) 997-8267

Parallax Serial LCD

2 rows x 16 characters, Non-backlit, with Piezospeaker (#27976)
2 rows x 16 characters, Backlit, with Piezospeaker (#27977)
4 rows x 20 characters, Backlit, with Piezospeaker (#27979)

The Parallax Serial LCDs are very functional, low-cost liquid crystal displays that can be easily interfaced to and controlled by a microcontroller using a I/O pin. The LCD displays provide basic text wrapping so that your text looks correct on the display. Full control over all of their advanced LCD features allows you to move the cursor anywhere on the display with a single instruction and turn the display on and off in any configuration. They support visible ASCII characters Dec 32-127, and in addition you may define up to eight of your own custom characters to display anywhere on the LCD.

NOTE: If your Serial LCD Display does not have a speaker on the back, use the specifications and information in the Product Change Notice: Revision E and Earlier section on page 11.

Features

- Clear 40-pixel characters (8 H x 5 W)
- Supports ASCII DEC characters 32-127
- Define up to eight custom characters
- Automatic text wrapping
- Single command cursor placement
- Single command clears the display
- Define up to eight custom characters
- Select 2400, 9600, or 19,200 baud with switches on the back of the device
- Display type: STN, YG, positive transflective LCD
- Adjustable contrast knob on the back of the device
- YG LED for backlit model displays

Key Specifications

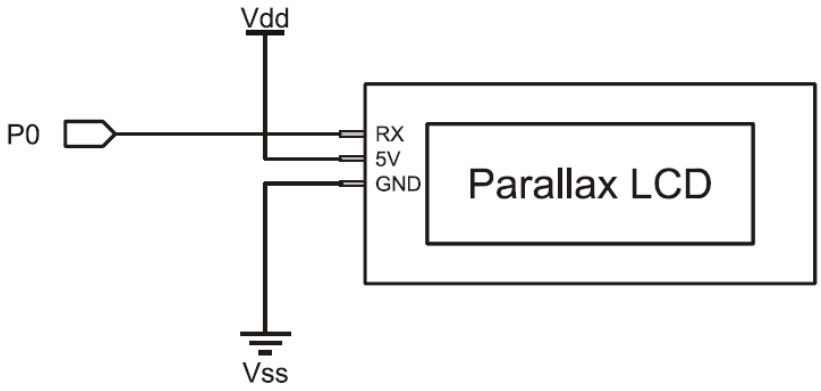
- Power requirements:
  - Non-backlit: +5 VDC, 20 mA
  - Backlit: +5 VDC, 20 mA (light off), ~ 80 mA typical (light on)
- Communication: Selectable asynchronous serial baud rates: 2400, 9600, 19200
- Operating temperature: -4 to +158 °F (-20 to +70°C)
- Dimensions: NOTE - Board and LCD size and style may vary
  - 2x16: Approx. 1.42 x 3.15 in (36 x 80 mm)
  - 4x20: Approx. 2.37 x 3.86 in (60.2 x 98.1 mm)

Quick-Start Circuit

The Serial LCDs should be powered from an external regulated 5 V power supply. Make sure the power supply has an adequate current rating to power the Serial LCD and the BASIC Stamp, Propeller chip, or whichever microcontroller and other devices you are using.

CAUTION

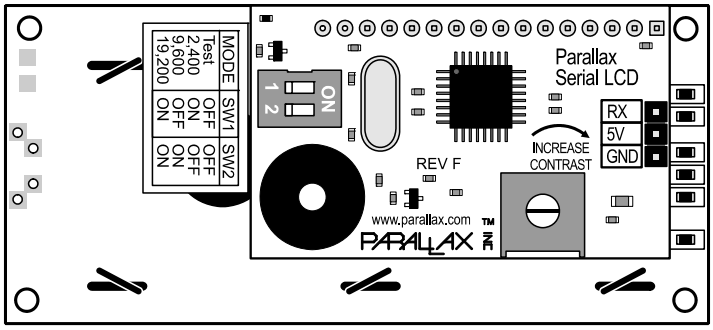
DO NOT PROVIDE A SIGNAL TO THE 'RX' PIN BEFORE APPLYING 5 VDC TO THE '5V' PIN.



Baud Rate Setup

After connecting the Serial LCD, you will need to select the baud rate at which you are going to send it data. You have three choices: 2400, 9600, and 19,200 baud. To set the baud rate, move the dip switches on the back of the LCD into the correct positions according to the table next to the switches, which is also repeated below:

MODE	SW1	SW2
Test	OFF	OFF
2,400	ON	OFF
9,600	OFF	ON
19,200	ON	ON



As you can see from the table, there is also a fourth choice called Test. Use this Test mode to confirm that the power and ground to the LCD are hooked up correctly before you send it any data. Move the dip switches to the Test setting and turn on the power. The LCD display should turn on with the backlight on (models 27977, 27979) and display the following text:

```
Parallax, Inc.
www.parallax.com
```

If you don't see the text at first, try adjusting the LCD contrast by turning the pot labeled "Increase Contrast" with a screwdriver. Turn it in the direction of the arrow to make the characters show up more clearly. If you still don't see the characters, go back and check your electrical connections and try again. Once you've successfully completed test mode, move the dip switches to the correct positions to select the baud rate you want to use for your application.

Displaying Text

Now that the LCD is set up, it's time to start sending text to the display. To display a character of text on the Serial LCD, simply send the ASCII code of that character to the Serial LCD over the serial port at the correct baud rate.

When a character is received, the Serial LCD displays that character at the current cursor position and then moves the cursor one position to the right. When you first turn on the LCD, the cursor is in the leftmost position on the top line, as you might expect. The short bar on the bottom of the character shows where the cursor is positioned currently.

Once you've sent a full line characters to the LCD, you will notice that the cursor automatically wraps around to the leftmost position of the second line, just like the text in a book. The text will wrap like this at the end of every line, with the end of the bottom line wrapping back around to the top line of the LCD. The text will never "run off" the display; you'll always see all of the characters you send.

Example code is provided below. You may download the example code files from the 27976, 27977, or 27979 product pages at www.parallax.com; just enter any of these product numbers in the "search" field on the home page

BASIC Stamp® 2 Example Code

You may download the example code from the 27976, 27977, or 27979 product pages at www.parallax.com; just enter any of these product numbers in the "search" field on the home page. Try the following code on your BASIC Stamp 2 to send a text string to the LCD display. First, set the baud rate on your Serial LCD to 19,200. Then, load the code below into your BASIC Stamp 2 and run it.

You will see the text string show up and wrap to the second line of the display.

In all of your Serial LCD code, you should pause for 100 ms at start-up to give time for the Serial LCD to initialize. You should also set the serial port pin on the BASIC Stamp to HIGH before the 100 ms start-up delay, as this is the normal state of a serial port when it isn't sending any data.

```
' {$STAMP BS2}
' {$PBASIC 2.5}

TxPin         CON    0
Baud19200     CON    32

  HIGH TxPin               ' Set pin high to be a serial port
  PAUSE 100                ' Pause for Serial LCD to initialize

  SEROUT TxPin, Baud19200, ["Hello, this text will wrap."]
```

Propeller P8X32A Example Code

Try the following code on your Propeller to send a text string to the LCD display. First, set the baud rate on your Serial LCD to 19,200. Then, load the code below into your Propeller and load RAM or EEPROM.

You will see the text string show up and wrap to the second line of the display.

In all of your Serial LCD code, you should wait for 100 ms after starting the FullDuplexSerial.spin object, to give the object and the Serial LCD time to initialize. The FullDuplexSerial.spin object is included with the Propeller Tool.

```
{{
Serial_LCD_Demo.spin
For Parallax Serial LCDs 27976, 27977, 27979
}}

CON
  _clkmode = xtal1 + pll16x
  _xinfreq = 5_000_000

  TX_PIN = 0
  BAUD = 19_200

OBJ

  LCD : "FullDuplexSerial.spin"

PUB Main

  LCD.start(TX_PIN, TX_PIN, %1000, 19_200)
  waitcnt(clkfreq / 100 + cnt)               ' Pause for FullDuplexSerial.spin to initialize
  LCD.str(string("Hello, this text will wrap."))
```

Moving the Cursor

When you send a character to the Serial LCD, it always displays at the current cursor position. There are a few different ways to move the cursor on the Serial LCD display. After each character you send, the cursor automatically moves over one position. Along with this, there is a standard set of cursor move commands including Backspace, Carriage Return, and Line Feed.

The Backspace/Left command (Dec 8) moves the cursor one place to the left and the Right command (Dec 9) moves the cursor one place to the right. These can be useful for moving the cursor around to overwrite existing text. These commands wrap to the next line of the display, if necessary. The Line Feed command (Dec 10) moves the cursor to the next line of the display without changing the horizontal position of the cursor. The Carriage Return command (Dec 13) also moves the cursor to the next line, but it moves the cursor to the leftmost position on that line as well. The Form Feed command (Dec 12) clears the entire display and moves the cursor to the leftmost position on line 0, just like when you first turn on the display. You will need to pause for 5mS in your code after sending the Form Feed command, to give the Serial LCD time to clear the display. Except for Form Feed, none of these move commands affects the characters on the display.

There are also direct move commands that you can use to move the cursor to any position on the display with a single command. The commands in the range Dec 128 to 143 and Dec 148 to 163 move the cursor to the 16 different positions on each of the two lines of the model 27976 and 27977 LCDs. The commands in the range Dec 128 to 207 move the cursor to the 20 different positions on each of the four lines of the model 27979 LCD.

Controlling the Display

You also have control over the various display modes of the Serial LCD. The display-off command (Dec 21) turns off the display so that all of the characters disappear. The characters aren't erased from the display, though, and you can even keep writing new characters to the display when it is turned off. A trick to make a lot of text show up all at once, even at a slow baud rate, is to turn off the display and then send all of your text. Then, when you turn the display on again, all of the text appears instantly.

The display-on commands (Dec 22 to 25) turn the display back on and also control whether you want to display the cursor and/or make the cursor character blink. The cursor is the short bar that shows up below the character at the current cursor position. The blink option makes that character blink on and off repeatedly. You can turn the cursor and blink options on or off, in any combination, as listed in the command set table. You can change the cursor and blink mode even if the display is already on; you don't need to turn it off and then back on again.

With models 27977 and 27979, you can also control the backlight of the display. The backlight lights up the display so that it is easier to see in the dark. There are commands to turn the backlight on (Dec 17) and off (Dec 18).

Custom Characters

The Serial LCD has the capability to store up to eight user-defined custom characters. The custom characters are stored in RAM and so they need to be redefined if you turn off the power. You can display the custom characters by sending the commands Dec 0 to 7, as shown in the command set table. The custom character will display at the current cursor position.

The custom characters are five pixels wide by eight pixels high. Each of the characters is stored as a series of eight data bytes where the low five bits of each byte represent a row of pixels in the character. The high three bits of each byte are ignored. A bit value of one turns that pixel on (i.e. makes it black). The bottom row of pixels is often left blank (all zeros) to make it easier to see the cursor.

To define a custom character, you will send a total of 9 bytes to the Serial LCD. The first byte needs to be a valid define-custom-character command (Dec 248 to 255) and must be followed by eight data bytes that define the pixels of the character. The Serial LCD will always use the next eight bytes it receives to set the pixels of the character. The data bytes define the character starting at the topmost row of pixels, as shown in the example code.

BASIC Stamp 2 Custom Character Example

Define a custom character using the code example below. First, set the baud rate on your Serial LCD to 19,200. Then, load the code below into your BASIC Stamp 2 and run it. You will see a diamond character appear on the screen.

```
' {$STAMP BS2}
' {$PBASIC 2.5}

TxPin         CON    0
Baud19200     CON    32

  HIGH TxPin                            ' Set pin high to be a serial port
  PAUSE 100                             ' Pause for Serial LCD to initialize

  SEROUT TxPin, Baud19200, [250]        ' Define custom character 2
                                        ' Now send the eight data bytes
  SEROUT TxPin, Baud19200, [%00000]     ' %00000 =
  SEROUT TxPin, Baud19200, [%00100]     ' %00100 = *
  SEROUT TxPin, Baud19200, [%01110]     ' %01110 = * * *
  SEROUT TxPin, Baud19200, [%11111]     ' %11111 = * * * * *
  SEROUT TxPin, Baud19200, [%01110]     ' %01110 = * * *
  SEROUT TxPin, Baud19200, [%00100]     ' %00100 = *
  SEROUT TxPin, Baud19200, [%00000]     ' %00000 =
  SEROUT TxPin, Baud19200, [%00000]     ' %00000 =
  SEROUT TxPin, Baud19200, [2]          ' Display the new custom character 2
```

Propeller™ P8X32A Example Code

Define a custom character using the code example below. First, set the baud rate on your Serial LCD to 19,200. Then, load the code below into your Propeller and load RAM or EEPROM. You will see a diamond character appear on the screen. Note: the FullDuplexSerial.spin object is included with the Propeller Tool software.

```
{{
Serial_LCD_Custom_Character.spin
For Parallax Serial LCDs 27976, 27977, 27979
}}

CON
  _clkmode = xtal1 + pll16x
  _xinfreq = 5_000_000

  TX_PIN = 0
  BAUD = 19_200

OBJ

  LCD : "FullDuplexSerial.spin"


PUB Main
  LCD.start(TX_PIN, TX_PIN, %1000, 19_200)
  waitcnt(clkfreq / 100 + cnt)               ' Pause for FullDuplexSerial.spin to initialize


  LCD.tx(250)                                ' Define custom character 2
                                             ' Now send the eight data bytes
  LCD.tx(%00000)                             ' %00000 =
  LCD.tx(%00100)                             ' %00100 = *
  LCD.tx(%01110)                             ' %01110 = * * *
  LCD.tx(%11111)                             ' %11111 = * * * * *
  LCD.tx(%01110)                             ' %01110 = * * *
  LCD.tx(%00100)                             ' %00100 = *
  LCD.tx(%00000)                             ' %00000 =
  LCD.tx(%00000)                             ' %00000 =
  LCD.tx(2)                                  ' Display the new custom character 2
```

Playing Music

The Serial LCD has a built-in piezoelectric speaker which can play musical notes. The LCD includes a sophisticated music player enabling users to program songs and tunes. You can play musical notes by sending three types of note commands (Dec 214 to 232), as shown in the command set table.

Set Length

The “set length” commands (Dec 208 to 214) determines the length of time each note will play for. This value remains the same until another “set length” command is received. A note’s length can range from a 1/64th note to a whole note. A whole note is 2 seconds long.

Set Scale

The "set scale" commands (Dec 215 to 219) determines the octave each note will play in. This value remains the same until another "set scale" command is received. The current octave can be set to a value between 3 and 7, and each octave consists of 12 notes. The frequency for each note and scale combination is shown in the table below.

Scale	A	A#	B	C	C#	D	D#	E	F	F#	G	G#
3	220	233	247	262	277	294	311	330	349	370	392	415
4	440	466	494	523	554	587	622	659	698	740	784	831
5	880	932	988	1047	1109	1175	1245	1319	1397	1480	1568	1661
6	1760	1865	1976	2093	2217	2349	2489	2637	2794	2960	3136	3322
7	3520	3729	3951	4186	4435	4699	4978	5274	5588	5920	6272	6645

Play Note

The "play note" commands (Dec 220 to 232) play a single note for the currently set time and in the currently set octave.

Command Set

The tables on the following pages list all of the valid Serial LCD commands. Commands marked as N/A are invalid and are ignored. The lines of the LCD display are numbered starting from 0, with line 0 being the top line. The character positions on each line are numbered starting from 0, with position 0 being the leftmost position on the line.

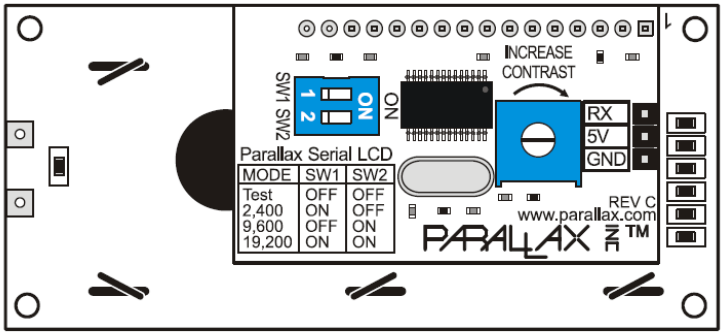
Dec	Hex	Action
0	00	Display custom character 0
1	01	Display custom character 1
2	02	Display custom character 2
3	03	Display custom character 3
4	04	Display custom character 4
5	05	Display custom character 5
6	06	Display custom character 6
7	07	Display custom character 7
8	08	Backspace / Left - The cursor is moved one position to the left. The command doesn't erase the character.
9	09	Right - The cursor is moved one position to the right. The command doesn't erase the character.
10	0A	Line Feed - The cursor is moved down one line. For the two line LCD model, if on line 0 it goes to line 1. If on line 1, it wraps around to line 0. The horizontal position remains the same.
Dec	**Hex**	**Action**
11	0B	N/A
12	0C	Form Feed - The cursor is moved to position 0 on line 0 and the entire display is cleared. Users must pause 5mS after this command.
13	0D	Carriage Return – For the two line LCD model, if on line 0 the cursor is moved to position 0 on line 1. If on line 1, it wraps around to position 0 on line 0.
14 - 16	0E - 10	N/A
17	11	Turn backlight on (only on models 27977, 27979)

Dec	Hex	Action
18	12	Turn backlight off (Default)
19 - 20	13 - 14	N/A
21	15	Turn the display off
22	16	Turn the display on, with cursor off and no blink
23	17	Turn the display on, with cursor off and character blink
24	18	Turn the display on, with cursor on and no blink (Default)
25	19	Turn the display on, with cursor on and character blink
26 - 31	1A - 1F	N/A
32 - 127	20 - 7F	Display ASCII characters. See the ASCII character set table.
128	80	Move cursor to line 0, position 0
129	81	Move cursor to line 0, position 1
130	82	Move cursor to line 0, position 2
131	83	Move cursor to line 0, position 3
132	84	Move cursor to line 0, position 4
133	85	Move cursor to line 0, position 5
134	86	Move cursor to line 0, position 6
135	87	Move cursor to line 0, position 7
136	88	Move cursor to line 0, position 8
137	89	Move cursor to line 0, position 9
138	8A	Move cursor to line 0, position 10
139	8B	Move cursor to line 0, position 11
140	8C	Move cursor to line 0, position 12
141	8D	Move cursor to line 0, position 13
142	8E	Move cursor to line 0, position 14
143	8F	Move cursor to line 0, position 15
144	90	Move cursor to line 0, position 16 (only on model 27979)
145	91	Move cursor to line 0, position 17 (only on model 27979)
146	92	Move cursor to line 0, position 18 (only on model 27979)
147	93	Move cursor to line 0, position 19 (only on model 27979)
148	94	Move cursor to line 1, position 0
149	95	Move cursor to line 1, position 1
150	96	Move cursor to line 1, position 2
151	97	Move cursor to line 1, position 3
152	98	Move cursor to line 1, position 4
153	99	Move cursor to line 1, position 5
154	9A	Move cursor to line 1, position 6
155	9B	Move cursor to line 1, position 7
156	9C	Move cursor to line 1, position 8
Dec	**Hex**	**Action**
157	9D	Move cursor to line 1, position 9
158	9E	Move cursor to line 1, position 10
159	9F	Move cursor to line 1, position 11
160	A0	Move cursor to line 1, position 12
161	A1	Move cursor to line 1, position 13
162	A2	Move cursor to line 1, position 14
163	A3	Move cursor to line 1, position 15

Dec	Hex	Action
164	A4	Move cursor to line 1, position 16 (only on model 27979)
165	A5	Move cursor to line 1, position 17 (only on model 27979)
166	A6	Move cursor to line 1, position 18 (only on model 27979)
167	A7	Move cursor to line 1, position 19 (only on model 27979)
168	A8	Move cursor to line 2, position 0 (only on model 27979)
169	A9	Move cursor to line 2, position 1 (only on model 27979)
170	AA	Move cursor to line 2, position 2 (only on model 27979)
171	AB	Move cursor to line 2, position 3 (only on model 27979)
172	AC	Move cursor to line 2, position 4 (only on model 27979)
173	AD	Move cursor to line 2, position 5 (only on model 27979)
174	AE	Move cursor to line 2, position 6 (only on model 27979)
175	AF	Move cursor to line 2, position 7 (only on model 27979)
176	B0	Move cursor to line 2, position 8 (only on model 27979)
177	B1	Move cursor to line 2, position 9 (only on model 27979)
178	B2	Move cursor to line 2, position 10 (only on model 27979)
179	B3	Move cursor to line 2, position 11 (only on model 27979)
180	B4	Move cursor to line 2, position 12 (only on model 27979)
181	B5	Move cursor to line 2, position 13 (only on model 27979)
182	B6	Move cursor to line 2, position 14 (only on model 27979)
183	B7	Move cursor to line 2, position 15 (only on model 27979)
184	B8	Move cursor to line 2, position 16 (only on model 27979)
185	B9	Move cursor to line 2, position 17 (only on model 27979)
186	BA	Move cursor to line 2, position 18 (only on model 27979)
187	BB	Move cursor to line 2, position 19 (only on model 27979)
188	BC	Move cursor to line 3, position 0 (only on model 27979)
189	BD	Move cursor to line 3, position 1 (only on model 27979)
190	BE	Move cursor to line 3, position 2 (only on model 27979)
191	BF	Move cursor to line 3, position 3 (only on model 27979)
192	C0	Move cursor to line 3, position 4 (only on model 27979)
193	C1	Move cursor to line 3, position 5 (only on model 27979)
194	C2	Move cursor to line 3, position 6 (only on model 27979)
195	C3	Move cursor to line 3, position 7 (only on model 27979)
196	C4	Move cursor to line 3, position 8 (only on model 27979)
197	C5	Move cursor to line 3, position 9 (only on model 27979)
198	C6	Move cursor to line 3, position 10 (only on model 27979)
199	C7	Move cursor to line 3, position 11 (only on model 27979)
200	C8	Move cursor to line 3, position 12 (only on model 27979)
201	C9	Move cursor to line 3, position 13 (only on model 27979)
202	CA	Move cursor to line 3, position 14 (only on model 27979)
Dec	**Hex**	**Action**
203	CB	Move cursor to line 3, position 15 (only on model 27979)
204	CC	Move cursor to line 3, position 16 (only on model 27979)
205	CD	Move cursor to line 3, position 17 (only on model 27979)
206	CE	Move cursor to line 3, position 18 (only on model 27979)
207	CF	Move cursor to line 3, position 19 (only on model 27979)
208	D0	Set note length to 1/64 note

Dec	Hex	Action
209	D1	Set note length to 1/32 note
210	D2	Set note length to 1/16 note
211	D3	Set note length to 1/8 note
212	D4	Set note length to 1/4 note
213	D5	Set note length to 1/2 note
214	D6	Set note length to whole note (2 seconds)
215	D7	Select the 3rd scale (A = 220 Hz)
216	D8	Select the 4th scale (A = 440 Hz)
217	D9	Select the 5th scale (A = 880 Hz)
218	DA	Select the 6th scale (A = 1760 Hz)
219	DB	Select the 7th scale (A = 3520 Hz)
220	DC	Play A note
221	DD	Play A# note
222	DE	Play B
223	DF	Play C
224	E0	Play C#
225	E1	Play D
226	E2	Play D#
227	E3	Play E
228	E4	Play F
229	E5	Play F#
230	E6	Play G
231	E7	Play G#
232	E8	Pause for current note length (no sound)
233 - 247	E9 – F7	N/A
248	F8	Define custom character 0. This command must be followed by eight data bytes.
249	F9	Define custom character 1. This command must be followed by eight data bytes.
250	FA	Define custom character 2. This command must be followed by eight data bytes.
251	FB	Define custom character 3. This command must be followed by eight data bytes.
252	FC	Define custom character 4. This command must be followed by eight data bytes.
253	FD	Define custom character 5. This command must be followed by eight data bytes.
254	FE	Define custom character 6. This command must be followed by eight data bytes.
255	FF	Define custom character 7. This command must be followed by eight data bytes.

Product Change Notice: Revision E and Earlier

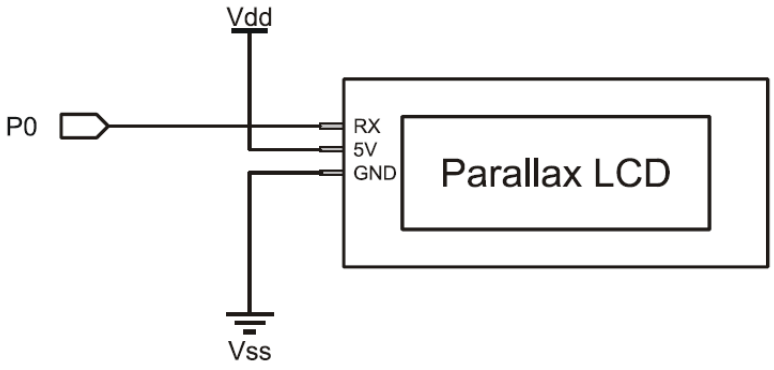


If your Serial LCD Display does not have a piezospeaker on the back and looks similar to the image above, please use the following information and specifications when using this product. All connections, test code and commands remain the same, except for commands controlling the piezospeaker.

Key Specifications

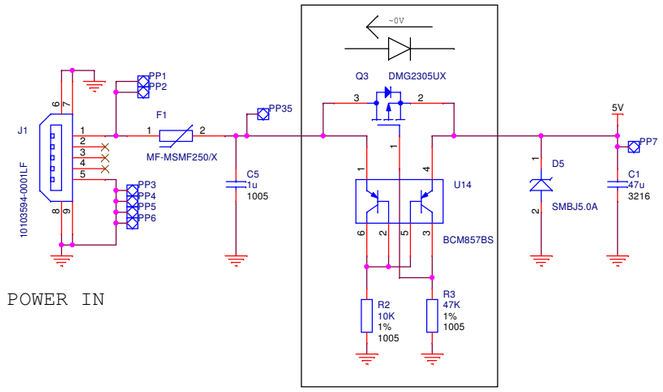
- Power requirements:
  - Non-backlit: +5 VDC, 20 mA
  - Backlit: +5 VDC, 20 mA (light off), 80 mA (light on)
- Communication: Selectable asynchronous serial baud rates: 2400, 9600, 19200
- Operating temperature: -4 to +158 °F (-20 to +70°C)
- Dimensions: **Board and LCD size and style may vary**
  - 2x16: Approx. 1.5 x 3.15 in (38 x 80 mm)
  - 4x20: Approx. 2.4 x 3.9 in (60 x 100 mm)

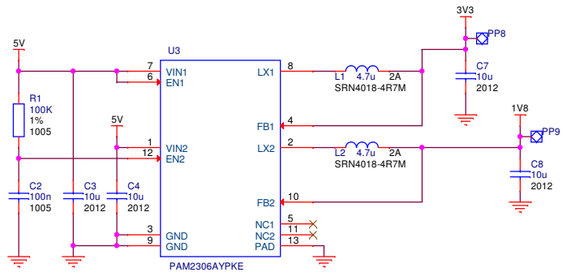
REV E CAUTION: DO NOT PROVIDE A SIGNAL TO THE 'RX' PIN BEFORE APPLYING 5 VDC TO THE '5V' PIN.

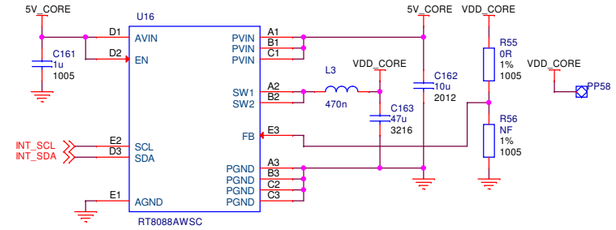


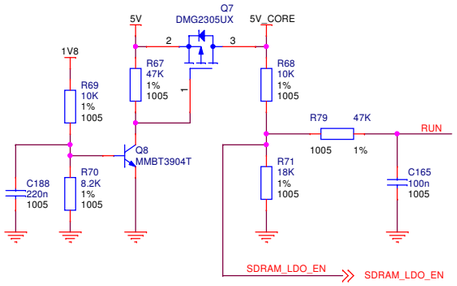
Revision History

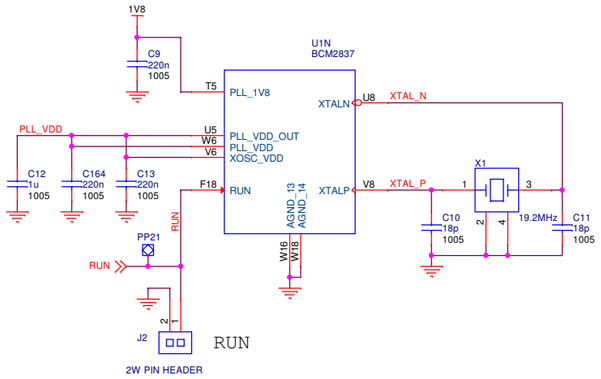
Version 3.1: corrected current draw specification on Page 1 from 50 mA to 80 mA.

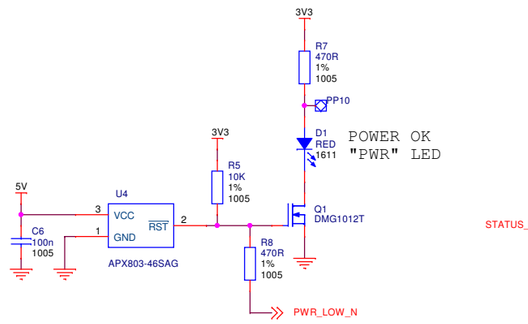


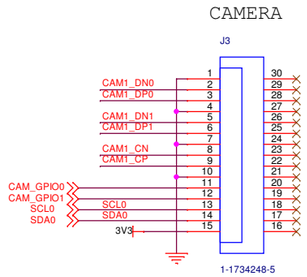


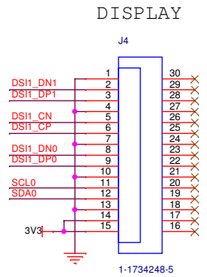


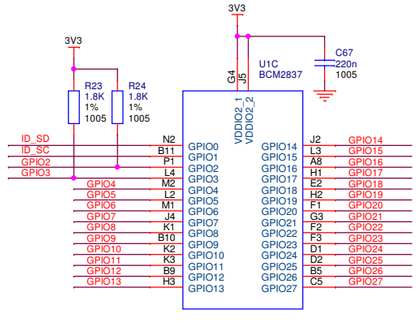


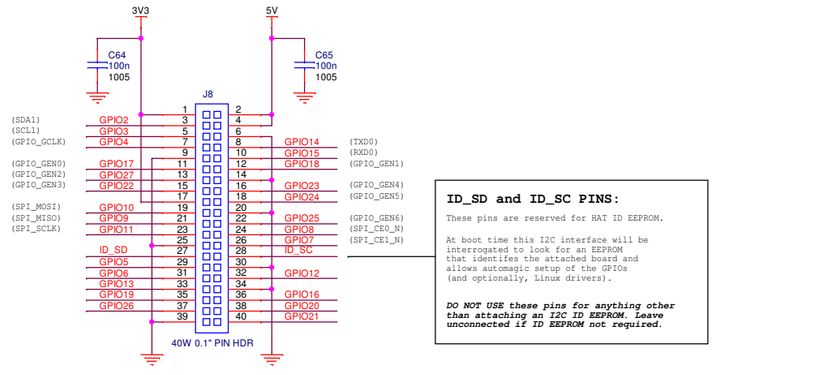










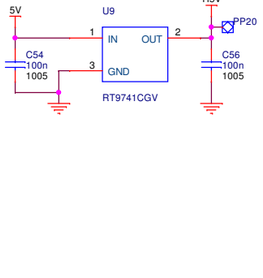


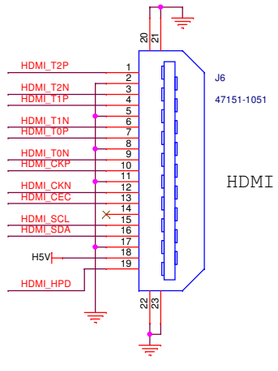
ID_SD and ID_SC PINS:

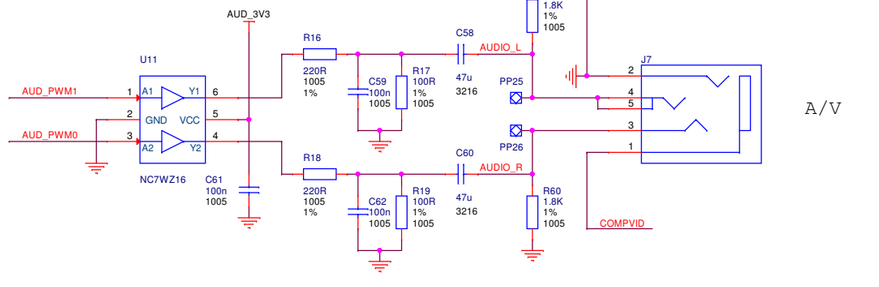
These pins are reserved for HAT ID EEPROM.

At boot time this I2C interface will be interrogated to look for an EEPROM that identifies the attached board and allows automagic setup of the GPIOs (and optionally, Linux drivers).

DO NOT USE these pins for anything other than attaching an I2C ID EEPROM. Leave unconnected if ID EEPROM not required.







Title: Raspberry Pi 3 Model B (Reduced Schematics)		Drawn By: James Adams	
Size: A2	Ref: RPI-3B		Rev: 1.2
Date: Monday, April 04, 2016	Sheet 1 of 1		

Data sheet

BNO055

Intelligent 9-axis absolute orientation sensor

Bosch Sensortec

BOSCH
Invented for life

BNO055: data sheet	
Document revision	1.2
Document release date	November 2014
Document number	BST-BNO055-DS000-12
Technical reference code(s)	0 273 141 209
Notes	Data in this document are subject to change without notice. Product photos and pictures are for illustration purposes only and may differ from the real product's appearance.

BNO055

INTELLIGENT ABSOLUTE ORIENTATION SENSOR, 9-AXIS SENSOR FUSION ALL-IN-ONE WINDOWS 8.x COMPLIANT SENSOR HUB

Basic Description

Key features:

- Outputs fused sensor data — Quaternion, Euler angles, Rotation vector, Linear acceleration, Gravity, Heading
- 3 sensors in one device — an advanced triaxial 16bit gyroscope, a versatile, leading edge triaxial 14bit accelerometer and a full performance geomagnetic sensor
- Small package — LGA package 28 pins, Footprint 3.8 x 5.2 mm², height 1.13 mm²
- Power Management — Intelligent Power Management: normal, low power and suspend mode available
- Common voltage supplies — V_{DD} voltage range: 2.4V to 3.6V
- Digital interface — HID-I2C (Windows 8 compatible), I²C, UART, V_{DDIO} voltage range: 1.7V to 3.6V
- Consumer electronics suite — MSL1, RoHS compliant, halogen-free, Operating temperature: -40°C ... +85°C

Key features of integrated sensors:

Accelerometer features

- Programmable functionality — Acceleration ranges ±2g/±4g/±8g/±16g, Low-pass filter bandwidths 1kHz - <8Hz, Operation modes:
  - Normal
  - Suspend
  - Low power
  - Standby
  - Deep suspend
- On-chip interrupt controller — Motion-triggered interrupt-signal generation for
  - any-motion (slope) detection
  - slow or no motion recognition
  - high-g detection

Gyroscope features

- Programmable functionality

  Ranges switchable from ±125°/s to ±2000°/s
  Low-pass filter bandwidths 523Hz - 12Hz
  Operation modes:
  - Normal
  - Fast power up
  - Deep suspend
  - Suspend
  - Advanced power save

- On-chip interrupt controller

  Motion-triggered interrupt-signal generation for
  - any-motion (slope) detection
  - high rate

Magnetometer features

- Flexible functionality

  Magnetic field range typical ±1300µT (x-, y-axis); ±2500µT (z-axis)
  Magnetic field resolution of ~0.3µT
  Operating modes:
  - Low power
  - Regular
  - Enhanced regular
  - High Accuracy

  Power modes:
  - Normal
  - Sleep
  - Suspend
  - Force

Typical applications

- Navigation
- Robotics
- Fitness and well-being
- Augmented reality
- Context awareness
- Tablets and ultra-books

General description

The BNO055 is a System in Package (SiP), integrating a triaxial 14-bit accelerometer, a triaxial 16-bit gyroscope with a range of ±2000 degrees per second, a triaxial geomagnetic sensor and a 32-bit cortex M0+ microcontroller running Bosch Sensortec sensor fusion software, in a single package.

The corresponding chip-sets are integrated into one single 28-pin LGA 3.8mm x 5.2mm x 1.1 mm housing. For optimum system integration the BNO055 is equipped with digital bi-directional I^2C and UART interfaces. The I^2C interface can be programmed to run with the HID-I2C protocol turning the BNO055 into a plug-and-play sensor hub solution for devices running the Windows 8.0 or 8.1 operating system.

Contents

Table of Figures

List of Tables

Specification

If not stated otherwise, the given values are over lifetime and full performance temperature and voltage ranges, minimum/maximum values are ±3 sigma.

1.1 Electrical specification

Table 0-1: Electrical parameter specification

OPERATING CONDITIONS BNO055						
Parameter	**Symbol**	**Condition**	**Min**	**Typ**	**Max**	**Unit**
Supply Voltage (only Sensors)	V_{DD}	--	2.4	--	3.6	V
Supply Voltage (µC and I/O Domain)	V_{DDIO}	--	1.7	--	3.6	V
Voltage Input Low Level (UART, I2C)	V_{DDIO_VIL}	V_{DDIO} = 1.7-2.7V	--	--	0.25	V_{DDIO}
		V_{DDIO} = 2.7-3.6V	--	--	0.3	V_{DDIO}
Voltage Input High Level (UART, I2C)	V_{DDIO_VIH}	V_{DDIO} = 1.7-2.7V	0.7	--	--	V_{DDIO}
		V_{DDIO} = 2.7-3.6V	0.55	--	--	V_{DDIO}
Voltage Output Low Level (UART, I2C)	V_{DDIO_VOL}	V_{DDIO} > 3V , I_{OL} =20mA	--	0.1	0.2	V_{DDIO}
Voltage Output High Level (UART, I2C)	V_{DDIO_VOH}	V_{DDIO} > 3V , I_{OH} =10mA	0.9	0.8	--	V_{DDIO}
POR Voltage threshold on VDDIO-IN rising	V_{DDIO_POT+}	V_{DDIO} falls at 1V/ms or slower	--	1.45	--	V
POR Voltage threshold on VDDIO-IN falling	V_{DDIO_POT-}		--	0.99	--	V
Operating Temperature	T_A	--	-40	--	+85	°C
Total supply current normal mode at T_A (9DOF @100Hz output data rate)	I_{DD} + I_{DDIO}	V_{DD} = 3V, V_{DDIO} = 2.5V	--	--	12.3	mA
Total supply current Low power mode at T_A	I_{DD_LPM}	V_{DD} = 3V, V_{DDIO} = 2.5V	--	--	0.4	mA
Total supply current suspend mode at T_A	I_{DD_SuM}	V_{DD} = 3V, V_{DDIO} = 2.5V	--	--	0.04	mA

1.2 Electrical and physical characteristics, measurement performance

Table 0-2: Electrical characteristics BNO055

OPERATING CONDITIONS BNO055						
Parameter	**Symbol**	**Condition**	**Min**	**Typ**	**Max**	**Unit**
Start-Up time	T_{Sup}	From Off to configuration mode		400		ms
POR time	T_{POR}	From Reset to Normal mode		650		ms
Data Rate	DR	s. Par. Fusion Output data rates				
Data rate tolerance 9DOF @100Hz output data rate (if internal oscillator is used)	DR_{tol}			±1		%

OPERATING CONDITIONS ACCELEROMETER						
Parameter	**Symbol**	**Condition**	**Min**	**Typ**	**Max**	**Units**
Acceleration Range	g_{FS2g}	Selectable via serial digital interface		±2		g
	g_{FS4g}			±4		g
	g_{FS8g}			±8		g
	g_{FS16g}			±16		g

OUTPUT SIGNAL ACCELEROMETER (ACCELEROMETER ONLY MODE)						
Parameter	**Symbol**	**Condition**	**Min**	**Typ**	**Max**	**Units**
Sensitivity	S	All g_{FSXg} Values, T_A=25°C		1		LSB/mg
Sensitivity tolerance	S_{tol}	Ta=25°C, g_{FS2g}		±1	±4	%
Sensitivity Temperature Drift	TCS	g_{FS2g}, Nominal V_{DD} supplies, Temp operating conditions		±0.03		%/K
Sensitivity Supply Volt. Drift	S_{VDD}	g_{FS2g}, T_A=25°C, $V_{DD\_min} \leq V_{DD} \leq V_{DD\_max}$		0.065	0.2	%/V
Zero-g Offset (x,y.z)	Off_{xyz}	g_{FS2g}, T_A=25°C, nominal V_{DD} supplies, over life-time	-150	±80	+150	mg
Zero-g Offset Temperature Drift	TCO	g_{FS2g}, Nominal V_{DD} supplies		±1	+/-3.5	mg/K
Zero-g Offset Supply Volt. Drift	Off_{VDD}	g_{FS2g}, T_A=25°C, $V_{DD\_min} \leq V_{DD} \leq V_{DD\_max}$		1.5	2.5	mg/V
Bandwidth	bw_8	2nd order filter, bandwidth programmable		8		Hz
	bw_{16}			16		Hz
	bw_{31}			31		Hz
	bw_{63}			63		Hz
	bw_{125}			125		Hz
	bw_{250}			250		Hz
	bw_{500}			500		Hz

	bw_{1000}			1,000		Hz
Nonlinearity	NL	best fit straight line, g_{FS2g}		0.5	2	%FS
Output Noise Density	n_{rms}	g_{FS2g}, T_A=25°C Nominal V_{DD} supplies Normal mode		150	190	µg/√Hz

MECHANICAL CHARACTERISTICS ACCELEROMETER

Parameter	Symbol	Condition	Min	Typ	Max	Units
Cross Axis Sensitivity	CAS	relative contribution between any two of the three axes		1	2	%
Alignment Error	E_A	relative to package outline		0.5	2	°

OPERATING CONDITIONS GYROSCOPE

Parameter	Symbol	Condition	Min	Typ	Max	Unit
Rate Range	R_{FS125}	Selectable via serial digital interface		125		°/s
	R_{FS250}			250		°/s
	R_{FS500}			500		°/s
	R_{FS1000}			1,000		°/s
	R_{FS2000}			2,000		°/s

OUTPUT SIGNAL GYROSCOPE (GYRO ONLY MODE)

Sensitivity via register Map	S	Ta=25°C		16.0 900		LSB/°/s rad/s
Sensitivity tolerance	S_{tol}	Ta=25°C, R_{FS2000}	--	±1	±3	%
Sensitivity Change over Temperature	TCS	Nominal V_{DD} supplies -40°C ≤ T_A ≤ +85°C R_{FS2000}		±0.03	±0.07	%/K
Sensitivity Supply Volt. Drift	S_{VDD}	T_A=25°C, V_{DD_min} ≤ V_{DD} ≤ V_{DD_max}		<0.4		%/V
Nonlinearity	NL	best fit straight line R_{FS1000}, R_{FS2000}		±0.05	±0.2	%FS
Zero-rate Offset	Off Ω_x Ω_y and Ω_z	Nominal V_{DD} supplies T_A=25°C, Slow and fast offset cancellation off	-3	±1	+3	°/s
Zero-Ω Offset Change over Temperature	TCO	Nominal V_{DD} supplies -40°C ≤ T_A ≤ +85°C R_{FS2000}		±0.015	±0.03	°/s per K
Zero-Ω Offset Supply Volt. Drift	OffΩ_{VDD}	T_A=25°C, V_{DD_min} ≤ V_{DD} ≤ V_{DD_max}		0.1		°/s /V
Output Noise	n_{rms}	rms, BW=47Hz (@ 0.014°/s/√Hz)		0.1	0.3	°/s

Bandwidth BW	f_{-3dB}			523 230 116 64 47 32 23 12		Hz

MECHANICAL CHARACTERISTICS GYROSCOPE

Cross Axis Sensitivity	CAS	Sensitivity to stimuli in non-sense-direction		±1	±3	%

OPERATING CONDITIONS MAGNETOMETER (MAGNETOMETER ONLY MODE)

Parameter	Symbol	Condition	Min	Typ	Max	Units
Magnetic field range[1]	Brg,xy	TA=25°C	±1200	±1300		µT
	Brg,z		±2000	±2500		µT
Magnetometer heading accuracy[2]	As heading	30µT horizontal geomagnetic field component, TA=25°C			±2.5	deg

MAGNETOMETER OUTPUT SIGNAL

Parameter	Symbol	Condition	Min	Typ	Max	Unit
Device Resolution	$D_{res,m}$	T_A=25°C		0.3		µT
Gain error[3]	$G_{err,m}$	After API compensation T_A=25°C Nominal V_{DD} supplies		±5	±8	%
Sensitivity Temperature Drift	TCS_m	After API compensation -40°C ≤ T_A ≤ +85°C Nominal V_{DD} supplies		±0.01	±0.03	%/K
Zero-B offset	OFF_m	T_A=25°C		±40		µT
Zero-B offset[4]	$OFF_{m,cal}$	After calibration in fusion mode -40°C ≤ T_A ≤ +85°C		±2		µT
Zero-B offset Temperature Drift	TCO_m	-40°C ≤ T_A ≤ +85°C		±0.23	±0.37	µT/K
Full-scale Nonlinearity	$NL_{m,FS}$	best fit straight line			1	%FS

[1] Full linear measurement range considering sensor offsets.
[2] The heading accuracy depends on hardware and software. A fully calibrated sensor and ideal tilt compensation are assumed.
[3] Definition: *gain error = ((measured field after API compensation) / (applied field))* – 1
[4] Magnetic zero-B offset assuming calibration in fusion mode. Typical value after applying calibration movements containing various device orientations (typical device usage).

Output Noise	$n_{rms,lp,m,xy}$	Low power preset x, y-axis, T_A=25°C Nominal V_{DD} supplies		1.0		µT
	$n_{rms,lp,m,z}$	Low power preset z-axis, T_A=25°C Nominal V_{DD} supplies		1.4		µT
	$n_{rms,rg,m}$	Regular preset T_A=25°C Nominal V_{DD} supplies		0.6		µT
	$n_{rms,eh,m}$	Enhanced regular preset T_A=25°C Nominal V_{DD} supplies		0.5		µT
	$n_{rms,ha,m}$	High accuracy preset T_A=25°C Nominal V_{DD} supplies		0.3		µT
Power Supply Rejection Rate	$PSRR_m$	T_A=25°C Nominal V_{DD} supplies		±0.5		µT/V

2. **Absolute Maximum Ratings**

Table 2-1: Absolute maximum ratings (preliminary target values)

Parameter	Symbol	Condition	Min	Max	Units
Voltage at Supply Pin	V_{DD} Pin		-0.3	4.2	V
	V_{DDIO} Pin		-0.3	3.6	V
Voltage at any Logic Pin	$V_{non-supply}$ Pin		-0.3	V_{DDIO}+0.3	V
Passive Storage Temp. Range	Trps	≤ 65% rel. H.	-50	+150	°C
Mechanical Shock	$MechShock_{200\mu s}$	Duration ≤ 200µs		10,000	g
	$MechShock_{1ms}$	Duration ≤ 1.0ms		2,000	g
	$MechShock_{freefall}$	Free fall onto hard surfaces		1.8	m
ESD	ESD_{HBM}	HBM, at any Pin		2	kV
	ESD_{CDM}	CDM		400	V
	ESD_{MM}	MM		200	V

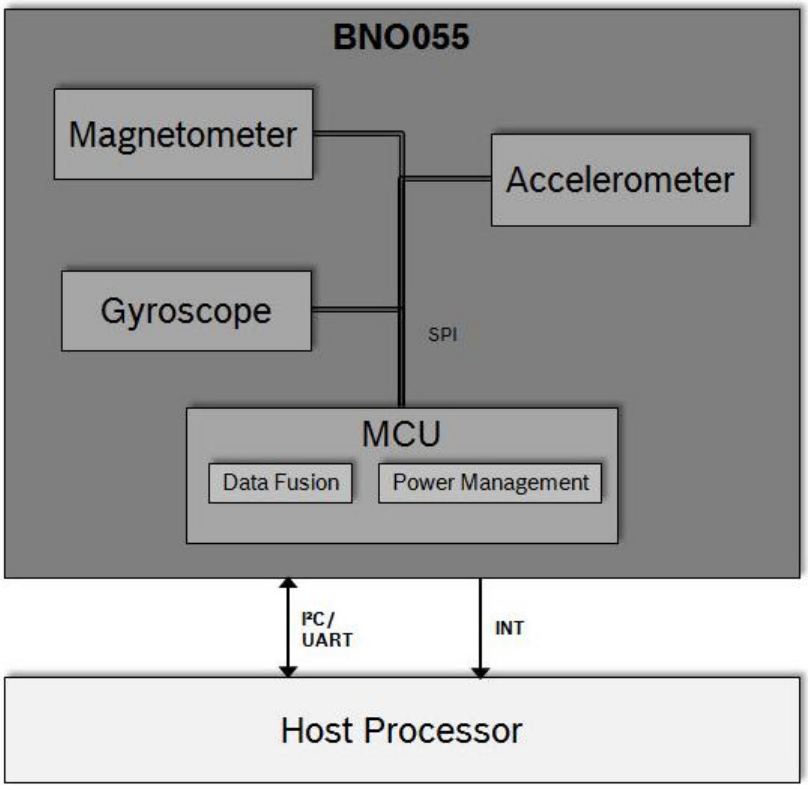
Note:
Stress above these limits may cause damage to the device. Exceeding the specified electrical limits may affect the device reliability or cause malfunction.

3. Functional Description

3.1 Architecture

The following figure shows the basic building blocks of the BNO055 device.

Figure 1: system architecture



3.2 Power management

The BNO055 has two distinct power supply pins:

- V_{DD} is the main power supply for the internal sensors
- V_{DDIO} is a separate power supply pin used for the supply of the µC and the digital interfaces

For the switching sequence of power supply V_{DD} and V_{DDIO} it is mandatory that V_{DD} is powered on and driven to the specified level before or at the same time as V_{DDIO} is powered ON. Otherwise there are no limitations on the voltage levels of both pins relative to each other, as long as they are used within the specified operating range.

The sensor features a power-on reset (POR), initializing the register map with the default values and starting in CONFIG mode. The POR is executed at every power on and can also be triggered either by applying a low signal to the nRESET pin for at least 20ns or by setting the RST_SYS bit in the SYS_TRIGGER register.

The BNO055 can be configured to run in one of the following power modes: normal mode, low power mode, and suspend mode. These power modes are described in more detail in section Power Modes

Power Modes
The BNO055 support three different power modes: Normal mode, Low Power Mode, and Suspend mode.
The power mode can be selected by writing to the PWR_MODE register as defined in the table below. As default at start-up the BNO055 will run in Normal mode.

Table 3-1: power modes selection

Parameter	Value	[Reg Addr]: Reg Value
Power Mode	Normal Mode	[PWR_MODE]: xxxxxx00b
	Low Power Mode	[PWR_MODE]: xxxxxx01b
	Suspend Mode	[PWR_MODE]: xxxxxx10b

3.2.1 **Normal Mode**

In normal mode all sensors required for the selected operating mode (see section 3.3) are always switched ON. The register map and the internal peripherals of the MCU are always operative in this mode.

3.2.2 **Low Power Mode**

If no activity (i.e. no motion) is detected for a configurable duration (default 5 seconds), the BNO055 enters the low power mode. In this mode only the accelerometer is active. Once motion is detected (i.e. the accelerometer signals an any-motion interrupt), the system is woken up and normal mode is entered. The following settings are possible.

Table 3-2: Low power modes - Interrupts

Description	Parameter	Value	Reg Value	Restriction
Entering to sleep: NO Motion Interrupt	Detection Type	No Motion	[ACC_NM_SET] : xxxxxx1b	n/a
		Detection Axis	[ACC_INT_Settings] : bit4-bit2	Shares common bit with Any Motion interrupt axis selection
	Params	Duration	[ACC_NM_SET] : bit6-bit1	n/a
		Threshold	[ACC_NM_THRE] : bit7-bit0	n/a

Description	Parameter	Value	Reg Value
Waking up: Any Motion Interrupt	Detection Type	Detection Axis	[ACC_INT_Settings] : bit4-bit2
	Params	Duration	[ACC_INT_Settings] : bit1-bit0
		Threshold	[ACC_AM_THRES] : bit7-bit0

Additionally, the interrupt pins can also be configured to provide HW interrupt to the host.

The BNO055 is by default configured to have optimum values for entering into sleep and waking up. To restore these values, trigger system reset by setting RST_SYS bit in SYS_TRIGGER register.

There are some limitations to achieve the low power mode performance:

- Only No and Any motion interrupts are applicable and High-G and slow motion interrupts are not applicable in low power mode.
- Low power mode is not applicable where accelerometer is not employed.

3.2.3 **Suspend Mode**

In suspend mode the system is paused and all the sensors and the microcontroller are put into sleep mode. No values in the register map will be updated in this mode. To exit from suspend mode the mode should be changed by writing to the PWR_MODE register (see Table 3-1).

3.3 Operation Modes

The BNO055 provides a variety of output signals, which can be chosen by selecting the appropriate operation mode. The table below lists the different modes and the available sensor signals.

Table 3-3: Operating modes overview

Operating Mode		Available sensor signals			Fusion Data	
		Accel	Mag	Gyro	Relative orientation	Absolute orientation
	CONFIGMODE	-	-	-	-	-
Non-fusionmodes	ACCONLY	X	-	-	-	-
	MAGONLY	-	X	-	-	-
	GYROONLY	-	-	X	-	-
	ACCMAG	X	X	-	-	-
	ACCGYRO	X	-	X	-	-
	MAGGYRO	-	X	X	-	-
	AMG	X	X	X	-	-
Fusion modes	IMU	X	-	X	X	-
	COMPASS	X	X	-	-	X
	M4G	X	X		X	-
	NDOF_FMC_OFF	X	X	X	-	X
	NDOF	X	X	X	-	X

The default operation mode after power-on is CONFIGMODE.

When the user changes to another operation mode, the sensors which are required in that particular sensor mode are powered, while the sensors whose signals are not required are set to suspend mode.

The BNO055 sets the following default settings for the sensors. The user can overwrite these settings in the register map when in CONFIGMODE.

Table 3-4: Default sensor settings

Sensor	Range	Bandwidth
Accelerometer	4G	62.5 Hz
Magnetometer	NA	10 Hz
Gyroscope	2000 dps	32 Hz

In any mode, the sensor data are available in the data register based on the unit selected. The axis of the data is configured based on the axis-remap register configuration.

The operating mode can be selected by writing to the OPR_MODE register, possible register values and the corresponding operating modes are shown in the table below.

Table 3-5: operating modes selection

Parameter	Value	[Reg Addr]: Reg Value
CONFIG MODE	CONFIGMODE	[OPR_MODE]: xxxx0000b
Non-Fusion Mode	ACCONLY	[OPR_MODE]: xxxx0001b
	MAGONLY	[OPR_MODE]: xxxx0010b
	GYROONLY	[OPR_MODE]: xxxx0011b
	ACCMAG	[OPR_MODE]: xxxx0100b
	ACCGYRO	[OPR_MODE]: xxxx0101b
	MAGGYRO	[OPR_MODE]: xxxx0110b
	AMG	[OPR_MODE]: xxxx0111b
Fusion Mode	IMU	[OPR_MODE]: xxxx1000b
	COMPASS	[OPR_MODE]: xxxx1001b
	M4G	[OPR_MODE]: xxxx1010b
	NDOF_FMC_OFF	[OPR_MODE]: xxxx1011b
	NDOF	[OPR_MODE]: xxxx1100b

Table 3-6 below shows the time required to switch between CONFIGMODE and the other operating modes.

Table 3-6: Operating mode switching time

From	To	Switching time
CONFIGMODE	Any operation mode	7ms
Any operation mode	CONFIGMODE	19ms

3.3.1 **Config Mode**

This mode is used to configure BNO, wherein all output data is reset to zero and sensor fusion is halted. This is the only mode in which all the writable register map entries can be changed. (Exceptions from this rule are the interrupt registers (INT and INT_MSK) and the operation mode register (OPR_MODE), which can be modified in any operation mode.)

As being said, this mode is the default operation mode after power-on or RESET. Any other mode must be chosen to be able to read any sensor data.

3.3.2 **Non-Fusion Modes**

3.3.2.1 ACCONLY

If the application requires only raw accelerometer data, this mode can be chosen. In this mode the other sensors (magnetometer, gyro) are suspended to lower the power consumption. In this mode, the BNO055 behaves like a stand-alone acceleration sensor.

3.3.2.1 MAGONLY

In MAGONLY mode, the BNO055 behaves like a stand-alone magnetometer, with acceleration sensor and gyroscope being suspended.

3.3.2.2 GYROONLY

In GYROONLY mode, the BNO055 behaves like a stand-alone gyroscope, with acceleration sensor and magnetometer being suspended.

3.3.2.3 ACCMAG

Both accelerometer and magnetometer are switched on, the user can read the data from these two sensors.

3.3.2.4 ACCGYRO

Both accelerometer and gyroscope are switched on; the user can read the data from these two sensors.

3.3.2.5 MAGGYRO

Both magnetometer and gyroscope are switched on, the user can read the data from these two sensors.

3.3.2.6 AMG (ACC-MAG-GYRO)

All three sensors accelerometer, magnetometer and gyroscope are switched on.

3.3.3 **Fusion modes**

Sensor fusion modes are meant to calculate measures describing the orientation of the device in space. It can be distinguished between non-absolute or relative orientation and absolute orientation. Absolute orientation means orientation of the sensor with respect to the earth and its magnetic field. In other words, absolute orientation sensor fusion modes calculate the direction of the magnetic north pole.
In non-absolute or relative orientation modes, the heading of the sensor can vary depending on how the sensor is placed initially.

All fusion modes provide the heading of the sensor as quaternion data or in Euler angles (roll, pitch and yaw angle). The acceleration sensor is both exposed to the gravity force and to accelerations applied to the sensor due to movement. In fusion modes it is possible to separate the two acceleration sources, and thus the sensor fusion data provides separately linear acceleration (i.e. acceleration that is applied due to movement) and the gravity vector.

3.3.3.1 IMU (Inertial Measurement Unit)

In the IMU mode the relative orientation of the BNO055 in space is calculated from the accelerometer and gyroscope data. The calculation is fast (i.e. high output data rate).

3.3.3.2 COMPASS

The COMPASS mode is intended to measure the magnetic earth field and calculate the geographic direction.
The earth magnetic field is a vector with the horizontal components x,y and the vertical z component. It depends on the position on the globe and natural iron occurrence. For heading calculation (direction of compass pointer) only the horizontal components x and y are used. Therefore the vector components of the earth magnetic field must be transformed in the horizontal plane, which requires the knowledge of the direction of the gravity vector. To summarize, the heading can only be calculated when considering gravity and magnetic field at the same time.

However, the measurement accuracy depends on the stability of the surrounding magnetic field. Furthermore, since the earth magnetic field is usually much smaller than the magnetic fields that occur around and inside electronic devices, the compass mode requires calibration (see chapter 3.10)

3.3.3.3 M4G (Magnet for Gyroscope)

The M4G mode is similar to the IMU mode, but instead of using the gyroscope signal to detect rotation, the changing orientation of the magnetometer in the magnetic field is used. Since the magnetometer has much lower power consumption than the gyroscope, this mode is less power consuming in comparison to the IMU mode. There are no drift effects in this mode which are inherent to the gyroscope.

However, as for compass mode, the measurement accuracy depends on the stability of the surrounding magnetic field.
For this mode no magnetometer calibration is required and also not available.

3.3.3.4 NDOF_FMC_OFF

This fusion mode is same as NDOF mode, but with the Fast Magnetometer Calibration turned ‘OFF’.

3.3.3.5 NDOF

This is a fusion mode with 9 degrees of freedom where the fused absolute orientation data is calculated from accelerometer, gyroscope and the magnetometer. The advantages of combining all three sensors are a fast calculation, resulting in high output data rate, and high robustness from magnetic field distortions. In this mode the Fast Magnetometer calibration is turned ON and thereby resulting in quick calibration of the magnetometer and higher output data accuracy. The current consumption is slightly higher in comparison to the NDOF_FMC_OFF fusion mode.

3.4 Axis remap

The device mounting position should not limit the data output of the BNO055 device. The axis of the device can be re-configured to the new reference axis.

Axis configuration byte: Register Address: **AXIS_MAP_CONFIG**

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reserved		Remapped Z axis value		Remapped Y axis value		Remapped X axis value	

There are two bits are used to configure the axis remap which will define in the following way,

Value	Axis Representation
00	X - Axis
01	Y - Axis
10	Z- Axis
11	Invalid

Also, when user try to configure the same axis to two or more then BNO055 will take this as invalid condition and previous configuration will be restored in the register map. The default value is: X Axis = X, Y Axis = Y and Z Axis = Z (AXIS_REMAP_CONFIG = 0x24).

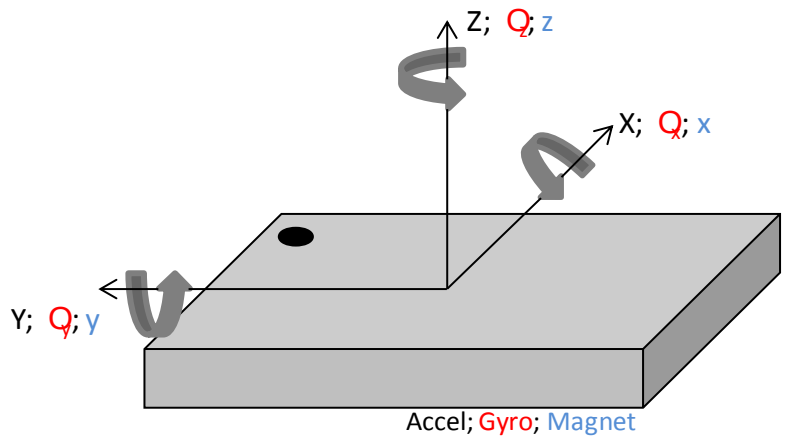
Axis sign configuration byte: Register Address: **AXIS_MAP_SIGN**

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reserved					Remapped X axis sign	Remapped Y axis sign	Remapped Z axis sign

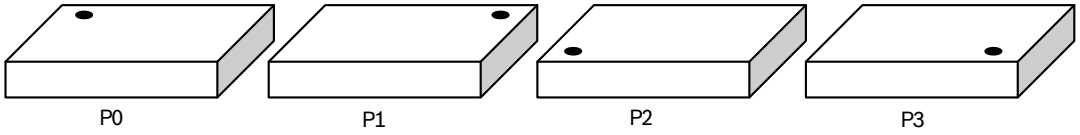
Value	Sign
0	Positive
1	Negative

The default value is 0x00.

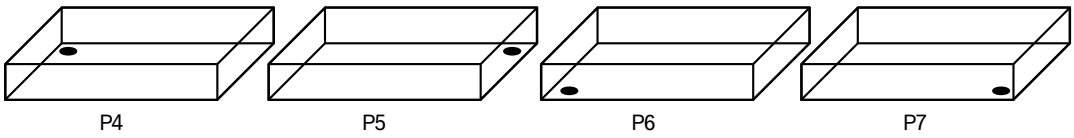
The default values correspond to the following coordinate system



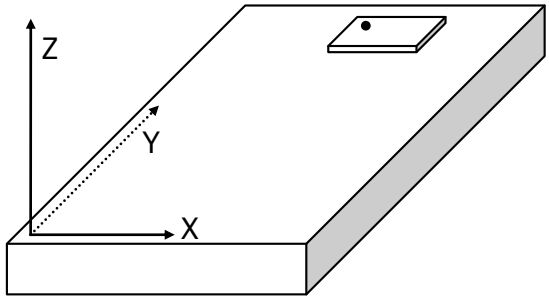
Some example placement for axis vs. register settings:



TOP VIEW



BOTTOM VIEW



P0

For the above described placements, following would be the axis configuration parameters.

Placement	AXIS_REMAP_CONFIG	AXIS_REMAP_SIGN
P0	0x21	0x04
P1 (default)	0x24	0x00
P2	0x24	0x06
P3	0x21	0x02
P4	0x24	0x03
P5	0x21	0x01
P6	0x21	0x07
P7	0x24	0x05

3.5 Sensor Configuration

The fusion outputs of the BNO055 are tightly linked with the sensor configuration settings. Due to this fact, the sensor configuration is limited when BNO055 is configured to run in any of the fusion operating mode. In any of the sensor modes the configuration settings can be updated by writing to the configuration registers as defined in the following sections.

3.5.1 **Default sensor configuration**

At power-on the sensors are configured with the default settings as defined in Table 3-8 below.

Table 3-7: Default sensor configuration at power-on

Sensors	Parameters	Value
Accelerometer	Power Mode	NORMAL
	Range	+/- 4g
	Bandwidth	62.5Hz
	Resolution	14 bits
Gyroscope	Power Mode	NORMAL
	Range	2000 °/s
	Bandwidth	32Hz
	Resolution	16 bits
Magnetometer	Power Mode	FORCED
	ODR	20Hz
	XY Repetition	15
	Z Repetition	16
	Resolution x/y/z	13/13/15 bits

3.5.2 **Accelerometer configuration**

The fusion outputs of the BNO055 are tightly linked with the accelerometer sensor settings. Therefore the configuration possibilities are restricted when running in any of the fusion operating modes. The accelerometer configuration can be changed by writing to the ACC_Config register, Table below shows different Accelerometer configurations

Table 3-8: Accelerometer configurations

Parameter	Values	[Reg Addr]: Reg Value	Restrictions
G Range	2G	[ACC_Config]: xxxxx00b	User selectable in all modes
	4G	[ACC_Config]: xxxxx01b	
	8G	[ACC_Config]: xxxxx10b	
	16G	[ACC_Config]: xxxxx11b	
Bandwidth	7.81Hz	[ACC_Config]: xxx000xxb	Auto controlled in fusion mode
	15.63Hz	[ACC_Config]: xxx001xxb	
	31.25Hz	[ACC_Config]: xxx010xxb	
	62.5Hz	[ACC_Config]: xxx011xxb	
	125Hz	[ACC_Config]: xxx100xxb	
	250Hz	[ACC_Config]: xxx101xxb	
	500Hz	[ACC_Config]: xxx110xxb	
	1000Hz	[ACC_Config]: xxx111xxb	
Operation Mode	Normal	[ACC_Config]: 000xxxxb	Auto controlled in fusion mode
	Suspend	[ACC_Config]: 001xxxxb	
	Low Power 1	[ACC_Config]: 010xxxxb	
	Standby	[ACC_Config]: 011xxxxb	
	Low Power 2	[ACC_Config]: 100xxxxb	
	Deep Suspend	[ACC_Config]: 101xxxxb	

The accelerometer sensor operation mode is not configurable by user when BNO power mode is configured as low power mode. BNO rewrites the user configured value to Normal mode when switching from config mode to any BNO operation mode. This used to achieve the BNO low power mode performance.

3.5.3 **Gyroscope configuration**

The fusion outputs of the BNO055 are tightly linked with the angular rate sensor settings. Therefore the configuration possibilities are restricted when running in any of the fusion operating modes. The gyroscope configuration can be changed by writing to the GYR_Config register, Table below shows different Gyroscope configurations

Table 3-9: Gyroscope configurations

Parameter	Values	[Reg Addr]: Register value	Restrictions
Range	2000 dps	[GYR_Config_0]: xxxx000b	Auto controlled in fusion mode
	1000 dps	[GYR_Config_0]: xxxx001b	
	500dps	[GYR_Config_0]: xxxx010b	
	250 dps	[GYR_Config_0]: xxxx011b	
	125 dps	[GYR_Config_0]: xxxx100b	
Bandwidth	523Hz	[GYR_Config_0]: xx000xxb	
	230Hz	[GYR_Config_0]: xx001xxb	
	116Hz	[GYR_Config_0]: xx010xxb	
	47Hz	[GYR_Config_0]: xx011xxb	
	23Hz	[GYR_Config_0]: xx100xxb	
	12Hz	[GYR_Config_0]: xx101xxb	
	64Hz	[GYR_Config_0]: xx110xxb	
	32Hz	[GYR_Config_0]: xx111xxb	
Operation Mode	Normal	[GYR_Config_1]: xxxx000b	
	Fast Power up	[GYR_Config_1]: xxxx001b	
	Deep Suspend	[GYR_Config_1]: xxxx010b	
	Suspend	[GYR_Config_1]: xxxx011b	
	Advanced Powersave	[GYR_Config_1]: xxxx100b	

3.5.4 **Magnetometer configuration**

The fusion outputs of the BNO055 are tightly linked with the magnetometer sensor settings. Therefore the configuration possibilities are restricted when running in any of the fusion operating modes. The magnetometer configuration can be changed by writing to the MAG_Config register, Table below shows different Magnetometer configurations.

Table 3-10: Magnetometer configurations

Parameter	Values	[Reg Addr]: Register value	Restrictions
Data output rate	2Hz	[MAG_Config]: xxxx000b	Auto controlled in fusion mode
	6Hz	[MAG_Config]: xxxx001b	
	8Hz	[MAG_Config]: xxxx010b	
	10Hz	[MAG_Config]: xxxx011b	
	15Hz	[MAG_Config]: xxxx100b	
	20Hz	[MAG_Config]: xxxx101b	
	25Hz	[MAG_Config]: xxxx110b	
	30Hz	[MAG_Config]: xxxx111b	
Operation Mode	Low Power	[MAG_Config]: xxx00xxb	
	Regular	[MAG_Config]: xxx01xxb	
	Enhanced Regular	[MAG_Config]: xxx10xxb	
	High Accuracy	[MAG_Config]: xxx11xxb	
Power Mode	Normal	[MAG_Config]: x00xxxxb	
	Sleep	[MAG_Config]: x01xxxxb	
	Suspend	[MAG_Config]: x10xxxxb	
	Force Mode	[MAG_Config]: x11xxxxb	

3.6 Output data

Depending on the selected operating mode the device will output either un-calibrated sensor data (in non-fusion mode) or calibrated / fused data (in fusion mode), this section describes the output data for each modes.

3.6.1 **Unit selection**

The measurement units for the various data outputs (regardless of operation mode) can be configured by writing to the UNIT_SEL register as described in Table 3-9.

Table 3-11: unit selection

Data	Units	[Reg Addr]: Register Value
Acceleration, Linear Acceleration, Gravity vector	m/s^2	[UNIT_SEL] : xxxxxx0b
	mg	[UNIT_SEL] : xxxxxx1b
Magnetic Field Strength	Micro Tesla	NA
Angular Rate	Dps	[UNIT_SEL] : xxxxx0xb
	Rps	[UNIT_SEL] : xxxxx1xb
Euler Angles	Degrees	[UNIT_SEL] : xxxx0xxb
	Radians	[UNIT_SEL] : xxxx1xxb
Quaternion	Quaternion units	NA
Temperature	°C	[UNIT_SEL] : xxx0xxxb
	°F	[UNIT_SEL] : xxx1xxxb

3.6.2 **Data output format**

The data output format can be selected by writing to the UNIT_SEL register, this allows user to switch between the orientation definition described by Windows and Android operating systems.

Table 3-12: Fusion data output format

Parameter	Values	[Reg Addr]: Register value
Fusion data output format	Windows	[UNIT_SEL]: 0xxxxxxb
	Android	[UNIT_SEL]: 1xxxxxxb

The output data format is based on the following convention regarding the rotation angles for roll, pitch and heading / yaw (compare also section 3.4):

Table 3-13: Rotation angle conventions

Rotation angle	Range (Android format)	Range (Windows format)
Pitch	+180° to -180° (turning clockwise decreases values)	-180° to +180° (turing clock-wise increases values)
Roll	-90° to +90° (increasing with increasing inclination)	
Heading / Yaw	0° to 360° (turning clockwise increases values)	

3.6.3 Fusion Output data rates

Table 3-14: Fusion output data rates

BNO055 Operating Mode	Data input rate			Algo calling rate	Data output rate			
	Accel	Mag	Gyro		Accel	Mag	Gyro	Fusion data
IMU	100Hz	NA	100Hz	100Hz	100Hz	NA	100Hz	100Hz
COMPASS	20Hz	20Hz	NA	20Hz	20Hz	20Hz	NA	20Hz
M4G	50Hz	50Hz	NA	50Hz	50Hz	50Hz	NA	50Hz
NDOF_FMC_OFF	100Hz	20Hz	100Hz	100Hz	100Hz	20Hz	100Hz	100Hz
NDOF	100Hz	20Hz	100Hz	100Hz	100Hz	20Hz	100Hz	100Hz

3.6.4 Sensor calibration data

The following section describes the register holding the calibration data of the sensors (see chapter 3.10). The offset and radius data can be read from these registers and stored in the host system, which could be later used to get the correct orientation data after 'Power on Reset' of the sensor.

3.6.4.1 Accelerometer offset

The accelerometer offset can be configured in the following registers, shown in the table below. There are 6 bytes required to configure the accelerometer offset (2 bytes for each of the 3 axis X, Y and Z). Configuration will take place only when the user writes the last byte (i.e., ACC_OFFSET_Z_MSB).

Table 3-15: Accelerometer Default-Reg settings

Reg Name	Default Reg Value (Bit 0 – Bit 7)
ACC_OFFSET_X_LSB	0x00
ACC_OFFSET_X_MSB	0x00
ACC_OFFSET_Y_LSB	0x00
ACC_OFFSET_Y_MSB	0x00
ACC_OFFSET_Z_LSB	0x00
ACC_OFFSET_Z_MSB	0x00

The range of the offsets varies based on the G-range of accelerometer sensor.

Table 3-16: Accelerometer G-range settings

Accelerometer G-range	Maximum Offset range in mg
2G	+/- 2000
4G	+/- 4000
8G	+/- 8000
16G	+/- 16000

Table 3-17: Accelerometer Unit settings

Unit	Representation
m/s^2	1 m/s^2 = 100 LSB
mg	1 mg = 1 LSB

3.6.4.2 Magnetometer offset

The magnetometer offset can be configured in the following registers,

Table 3-18: Magnetometer Default-Reg settings

Reg Name	Default Reg Value (Bit 0 – Bit 7)
MAG_OFFSET_X_LSB	0x00
MAG_OFFSET_X_MSB	0x00
MAG_OFFSET_Y_LSB	0x00
MAG_OFFSET_Y_MSB	0x00
MAG_OFFSET_Z_LSB	0x00
MAG_OFFSET_Z_MSB	0x00

There are 6 bytes required to configure the magnetometer offset (bytes (2 bytes for each of the 3 axis X, Y and Z). Configuration will take place only when the user writes the last byte (i.e., MAG_OFFSET_Z_MSB). Therefore the last byte must be written whenever the user wants to changes the configuration. The range of the magnetometer offset is +/-6400 in LSB.

Table 3-19: Magnetometer Unit settings

Unit	Representation
µT	1 µT = 16 LSB

3.6.4.3 Gyroscope offset

The gyroscope offset can be configured in the following registers, shown in the table below

Table 3-20: Gyroscope Default Reg-settings

Reg Name	Default Reg Value (Bit 0 – Bit 7)
GYR_OFFSET_X_LSB	0x00
GYR_OFFSET_X_MSB	0x00
GYR_OFFSET_Y_LSB	0x00
GYR_OFFSET_Y_MSB	0x00
GYR_OFFSET_Z_LSB	0x00
GYR_OFFSET_Z_MSB	0x00

There are 6 bytes required to configure the gyroscope offset (bytes (2 bytes for each of the 3 axis X, Y and Z). Configuration will take place only when the user writes the last byte (i.e., GYR_OFFSET_Z_MSB). Therefore the last byte must be written whenever the user wants to changes the configuration. The range of the offset varies based on the dps-range of gyroscope sensor.

Table 3-21: Gyroscope range settings

Gyroscope dps range	Maximum Offset range in LSB
2000	+/- 32000
1000	+/- 16000
500	+/- 8000
250	+/- 4000
125	+/- 2000

Table 3-22: Gyroscope unit settings

Unit	Representation
Dps	1 Dps = 16 LSB
Rps	1 Rps = 900 LSB

3.6.4.4 Radius

The radius of accelerometer, magnetometer and gyroscope can be configured in the following registers,

Table 3-23: Radius Default-Reg settings

Reg Name	Default Reg Value (Bit 0 – Bit 7)
ACC_RADIUS_LSB	0x00
ACC_RADIUS_MSB	0x00
MAG_RADIUS_LSB	0x00
MAG_RADIUS_MSB	0x00

There are 4 bytes (2 bytes for each accelerometer and magnetometer) to configure the radius. Configuration will take place only when user writes to the last byte (i.e., ACC_RADIUS_MSB and MAG_RADIUS_MSB). Therefore the last byte must be written whenever the user wants to changes the configuration. The range of the radius for accelerometer is +/-1000, magnetometer is +/-960 and Gyroscope is NA.

Table 3-24: Radius range settings

Radius for sensor	Maximum Range
Accelerometer	+/- 1000 LSB
Magnetometer	+/- 960 LSB

3.6.5 **Output data registers**

3.6.5.1 Acceleration data

In non-fusion mode uncompensated acceleration data for each axis X/Y/Z, can be read from the appropriate ACC_DATA_<axis>_LSB and ACC_DATA_<axis>_MSB registers.

In fusion mode the fusion algorithm output offset compensated acceleration data for each axis X/Y/Z, the output data can be read from the appropriate ACC_DATA_<axis>_LSB and ACC_DATA_<axis>_MSB registers. Refer table below for information regarding the data types for the acceleration data.

Table 3-25: Acceleration data

Parameter	Data type	bytes
Accel_Data_X	signed	2
Accel_Data_Y	signed	2
Accel_Data_Z	signed	2

3.6.5.2 Magnetic Field Strength

In non-fusion mode uncompensated field strength data for each axis X/Y/Z, can be read from the appropriate MAG_DATA_<axis>_LSB and MAG_DATA_<axis>_MSB registers.

In fusion mode the fusion algorithm output offset compensated magnetic field strength data for each axis X/Y/Z, the output data can be read from the appropriate MAG_DATA_<axis>_LSB and MAG_DATA_<axis>_MSB registers. Refer table below for information regarding the data types for the magnetic field strength.

Table 3-26: Magnetic field strength data

Parameter	Data type	bytes
Mag_Data_X	signed	2
Mag_Data_Y	signed	2
Mag_Data_Z	signed	2

3.6.5.3 Angular Velocity

In non-fusion mode uncompensated angular velocity (yaw rate) data for each axis X/Y/Z, can be read from the appropriate GYR_DATA_<axis>_LSB and GYR_DATA_<axis>_MSB registers.

In fusion mode the fusion algorithm output offset compensated angular velocity (yaw rate) data for each axis X/Y/Z, the output data can be read from the appropriate GYR_DATA_<axis>_LSB and GYR_DATA_<axis>_MSB registers. Refer table below for information regarding the data types for the angular velocity.

Table 3-27: Yaw rate data

Parameter	Data type	bytes
Gyr_Data_X	signed	2
Gyr_Data_Y	signed	2
Gyr_Data_Z	signed	2

3.6.5.4 Orientation (Euler angles)

Orientation output only available in fusion operation modes.

The fusion algorithm output offset and tilt compensated orientation data in Euler angles format for each DOF Heading/Roll/Pitch, the output data can be read from the appropriate EUL<dof>_LSB and EUL_<dof>_MSB registers. Refer table below for information regarding the data types and the unit representation for the Euler angle format.

Table 3-28: Compensated orientation data in Euler angles format

Parameter	Data type	bytes
EUL_Heading	Signed	2
EUL_Roll	Signed	2
EUL_Pitch	Signed	2

Table 3-29: Euler angle data representation

Unit	Representation
Degrees	1 degree = 16 LSB
Radians	1 radian = 900 LSB

3.6.5.5 Orientation (Quaternion)

Orientation output only available in fusion operating modes.

The fusion algorithm output offset and tilt compensated orientation data in quaternion format for each DOF w/x/y/z, the output data can be read from the appropriate QUA_DATA_<dof>_LSB and QUA_DATA_<dof>_MSB registers. Refer table below for information regarding the data types and the unit representation for the Orientation output.

Table 3-30: Compensated orientation data in quaternion format

Parameter	Data type	bytes
QUA_Data_w	Signed	2
QUA_Data_x	Signed	2
QUA_Data_y	Signed	2
QUA_Data_z	Signed	2

Table 3-31: Quaternion data representation

Unit	Representation
Quaternion (unit less)	1 Quaternion (unit less) = 2^14 LSB

3.6.5.6 Linear Acceleration

Linear acceleration output only available in fusion operating modes.

The fusion algorithm output linear acceleration data for each axis x/y/z, the output data can be read from the appropriate LIA_DATA_<axis>_LSB and LIA_DATA_<axis>_MSB registers. Refer table below for further information regarding the data types and the unit representation for Linear acceleration

Table 3-32: Linear Acceleration Data

Parameter	Data type	bytes
LIA_Data_X	signed	2
LIA_Data_Y	signed	2
LIA_Data_Z	signed	2

Table 3-33: Linear Acceleration data representation

Unit	Representation
m/s^2	1 m/s^2 = 100 LSB
mg	1 mg = 1 LSB

3.6.5.7 Gravity Vector

Gravity Vector output only available in fusion operating modes.

The fusion algorithm output gravity vector data for each axis x/y/z, the output data can be read from the appropriate GRV_DATA_<axis>_LSB and GRV_DATA_<axis>_MSB registers. Refer table below for further information regarding the data types and the unit representation for the Gravity vector.

Table 3-34: Gravity Vector Data

Parameter	Data type	bytes
GRV_Data_X	signed	2
GRV_Data_Y	signed	2
GRV_Data_Z	signed	2

Table 3-35: Gravity Vector data representation

Unit	Representation
m/s^2	1 m/s^2 = 100 LSB
mg	1 mg = 1 LSB

3.6.5.8 Temperature

The temperature output data can be read from the TEMP register. The table below describes the output data type and data representation (depending on selected unit).
The temperature can be read from one of two sources, the temperature source can be selected by writing to the TEMP_SOURCE register as detailed below.

Table 3-36: Temperature Data

Parameter	Data type	bytes
TEMP	signed	1

Table 3-37: Temperature data representation

Unit	Representation
°C	1°C = 1 LSB
F	2 F = 1 LSB

Table 3-38: Temperature Source Selection

Source	[Reg Addr]: Register Value
Accelerometer	[TEMP_SOURCE]: xxxxxx00b
Gyroscope	[TEMP_SOURCE]: xxxxxx01b

3.7 Interrupts

3.7.1 **Interrupt Pin**

INT is configured as interrupt pin for signaling an interrupt to the host. The interrupt trigger is configured as raising edge and is latched on to the INT pin. Once an interrupt occurs, the INT pin is set to high and will remain high until it is reset by host. This can be done by setting RST_INT in SYS_TRIGGER register.
Interrupts can be enabled by setting the corresponding bit in the interrupt enable register (INT_EN) and disabled when it is cleared.

Interrupt Pin Masking

Interrupts can be routed to the INT pin by setting the corresponding interrupt bit in the INT_MSK register.

Interrupt Status

Interrupt occurrences are stored in the interrupt status register (INT_STA). All bits in this register are cleared on read.

3.7.2 **Interrupt Settings**

3.7.2.1 Accelerometer Slow/No Motion Interrupt

The slow-motion/no-motion interrupt engine can be configured in two modes.

Slow-motion Interrupt is triggered when the measured slope of at least one enabled axis exceeds the programmable slope threshold for a programmable number of samples. Hence the engine behaves similar to the any-motion interrupt, but with a different set of parameters. In order to suppress false triggers, the interrupt is only generated (cleared) if a certain number *N* of consecutive slope data points is larger (smaller) than the slope threshold given by *slo_no_mot_dur<1:0>*. The number is *N = slo_no_mot_dur<1:0>* + 1.

In no-motion mode an interrupt is generated if the slope on all selected axes remains smaller than a programmable threshold for a programmable delay time. Figure 11 shows the timing diagram for the no-motion interrupt. The scaling of the threshold value is identical to that of the slow-motion interrupt. However, in no-motion mode register slo_no_mot_dur defines the delay time before the no-motion interrupt is triggered. Table 3-39 lists the delay times adjustable with register slo_no_mot_dur. The timer tick period is 1 second. Hence using short delay times can result in considerable timing uncertainty.

If bit *SM/NM* is set to ‘1’ (‘0’), the no-motion/slow-motion interrupt engine is configured in the no-motion (slow-motion) mode. Common to both modes, the engine monitors the slopes of the axes that have been enabled with bits *AM/NM_X_AXIS*, *AM/NM_Y_AXIS*, and *AM/NM_Z_AXIS* for the x-axis, y-axis and z-axis, respectively. The measured slope values are continuously compared against the threshold value defined in register ACC_NM_THRES. The scaling is such that 1 LSB of ACC_NM_THRES corresponds to 3.91 mg in 2g-range (7.81 mg in 4g-range, 15.6 mg in 8g-range and 31.3 mg in 16g-range). Therefore the maximum value is 996 mg in 2g-range (1.99g in 4g-range, 3.98g in 8g-range and 7.97g in 16g-range). The time difference between the successive acceleration samples depends on the selected bandwidth and equates to 1/(2 * bw).

Table 3-39: No-motion time-out periods

slo_no_mot_dur	Delay time	*slo_no_mot_dur*	*Delay time*	*slo_no_mot_dur*	*Delay Time*
0	1 s	16	40 s	32	88 s
1	2 s	17	48 s	33	96 s
2	3 s	18	56 s	34	104 s
...	...	19	64 s.	...	...
14	15 s	20	72 s	62	328 s
15	16 s	21	80 s	63	336 s

Note: slo_no_mot_dur values 22 to 31 are not specified

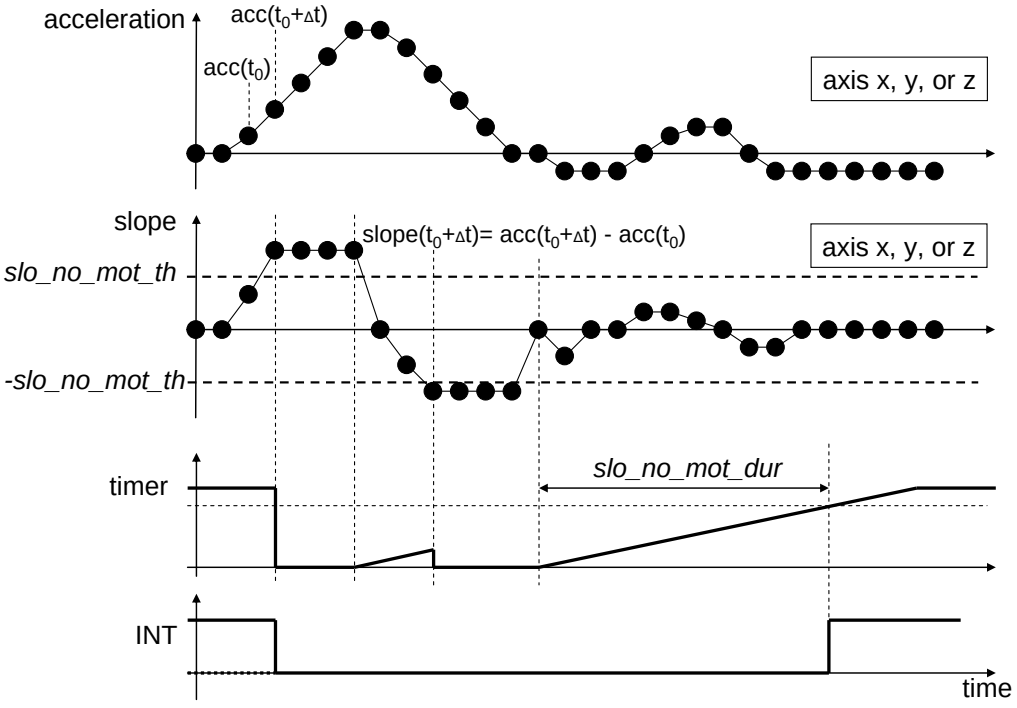


Table 3-40: Timing of No-motion interrupt

Params	Value	[Reg Addr]: Register Value
Detection Type	No Motion	[ACC_NM_SET]: xxxxxx0b
	Slow Motion	[ACC_NM_SET]: xxxxxx1b
Interrupt Parameters	Threshold	[ACC_NM_THRE]: bit7:bit0
	Duration	[ACC_NM_SET]: bit6:bit1
Axis selection	X-axis	[ACC_INT_Settings]: xxxx1xxb
	Y-axis	[ACC_INT_Settings]: xxx1xxxb
	Z-axis	[ACC_INT_Settings]: xx1xxxxb

3.7.2.2 Accelerometer Any Motion Interrupt

The any-motion interrupt uses the slope between successive acceleration signals to detect changes in motion. An interrupt is generated when the slope (absolute value of acceleration difference) exceeds a preset threshold. It is cleared as soon as the slope falls below the threshold. The principle is made clear in Figure 2: Principle of any-motion detection.

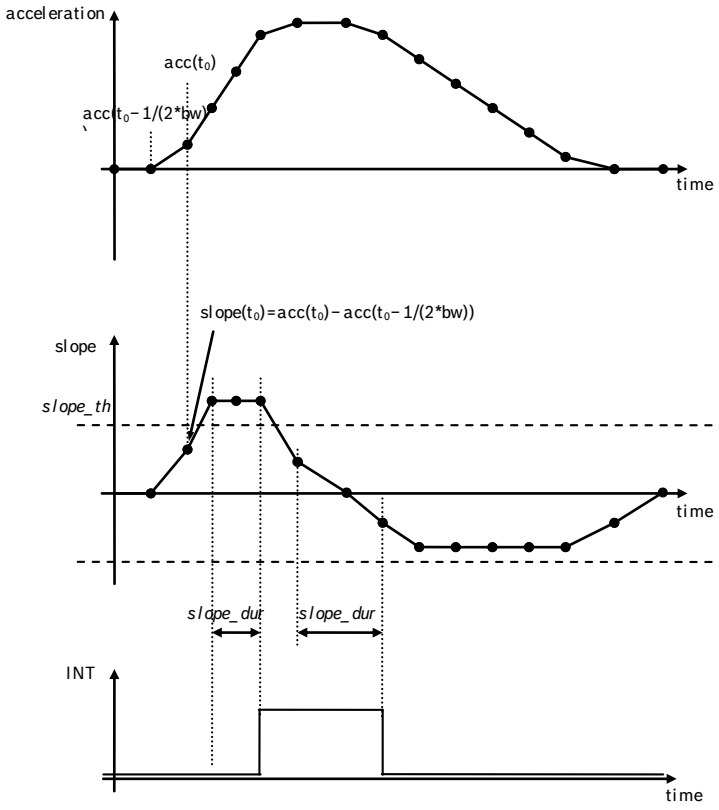


Figure 2: Principle of any-motion detection

The threshold is defined through register ACC_AM_THRES. In terms of scaling 1 LSB of ACC_AM_THRES corresponds to 3.91 mg in 2g-range (7.81 mg in 4g-range, 15.6 mg in 8g-range and 31.3 mg in 16g-range). Therefore the maximum value is 996 mg in 2g-range (1.99g in 4g-range, 3.98g in 8g-range and 7.97g in 16g-range).

The time difference between the successive acceleration signals depends on the selected bandwidth and equates to 1/(2*bandwidth) ()t=1/(2*bw)). In order to suppress false triggers, the interrupt is only generated (cleared) if a certain number *N* of consecutive slope data points is larger (smaller) than the slope threshold given by ACC_AM_THRES. This number is set by the AM_DUR bits. It is N = AM_DUR + 1.

Example: *AM_DUR* = 00b, ..., 11b = 1decimal, ..., 4decimal.

Enabling (disabling) for each axis:

Any-motion detection can be enabled (disabled) for each axis separately by writing ´1´ (´0´) to bits AM/NM_X_AXIS, AM/NM_Y_AXIS, AM/NM_Z_AXIS. The criteria for any-motion detection are fulfilled and the slope interrupt is generated if the slope of any of the enabled axes exceeds the threshold ACC_AM_THRES for [AM_DUR +1] consecutive times. As soon as the slopes of all enabled axes fall or stay below this threshold for [AM_DUR +1] consecutive times the interrupt is cleared unless interrupt signal is latched.

Table 3-41: Any-motion Interrupt parameters and Axis selection

Params	Value	[Reg Addr]: Register Value
Interrupt Parameters	Threshold	[ACC_AM_THRES]: bit7:bit0
	Duration	[ACC_INT_Settings]: bit1:bit0
Axis selection	X-axis	[ACC_INT_Settings]: xxxx1xxb
	Y-axis	[ACC_INT_Settings]: xxx1xxxb
	Z-axis	[ACC_INT_Settings]: xx1xxxxb

3.7.2.3 Accelerometer High G Interrupt

This interrupt is based on the comparison of acceleration data against a high-g threshold for the detection of shock or other high-acceleration events.

The high-g interrupt is enabled (disabled) per axis by writing ´1´ (´0´) to bits ACC_HIGH_G in the INT_EN register and enabling the axis in with bits HG_X_AXIS, HG_Y_AXIS, and HG_Z_AXIS, respectively in the ACC_INT_Settings register. The high-g threshold is set through the ACC_HG_THRES register. The meaning of an LSB of ACC_HG_THRES depends on the selected g-range: it corresponds to 7.81 mg in 2g-range, 15.63 mg in 4g-range, 31.25 mg in 8g-range, and 62.5 mg in 16g-range (i.e. increment depends from g-range setting).

The high-g interrupt is generated if the absolute value of the acceleration of at least one of the enabled axes (´or´ relation) is higher than the threshold for at least the time defined by the ACC_HG_DURATION register. The interrupt is reset if the absolute value of the acceleration of all enabled axes (´and´ relation) is lower than the threshold for at least the time defined by the ACC_HG_DURATION register. The interrupt status is stored in bit ACC_HIGH_G in the INT_STA register. The relation between the content of ACC_HG_DURATION and the actual delay of the interrupt generation is delay [ms] = [ACC_HG_DURATION + 1] * 2 ms. Therefore, possible delay times range from 2 ms to 512 ms.

Table 3-42: High-G Interrupt parameters and Axis selection

Params	Value	[Reg Addr]: Register Value
Interrupt Parameters	Threshold	[ACC_HG_THRES]: bit7 : bit0
	Duration	[ACC_HG_DURATION]: bit7 : bit0
Axis selection	X-axis	[ACC_INT_Settings]: xx1xxxxb
	Y-axis	[ACC_INT_Settings]: x1xxxxxb
	Z-axis	[ACC_INT_Settings]: 1xxxxxxb

3.7.2.4 Gyroscope High Rate Interrupt

This interrupt is based on the comparison of angular rate data against a high-rate threshold for the detection of shock or other high-angular rate events. The principle is made clear in Figure 3 below:

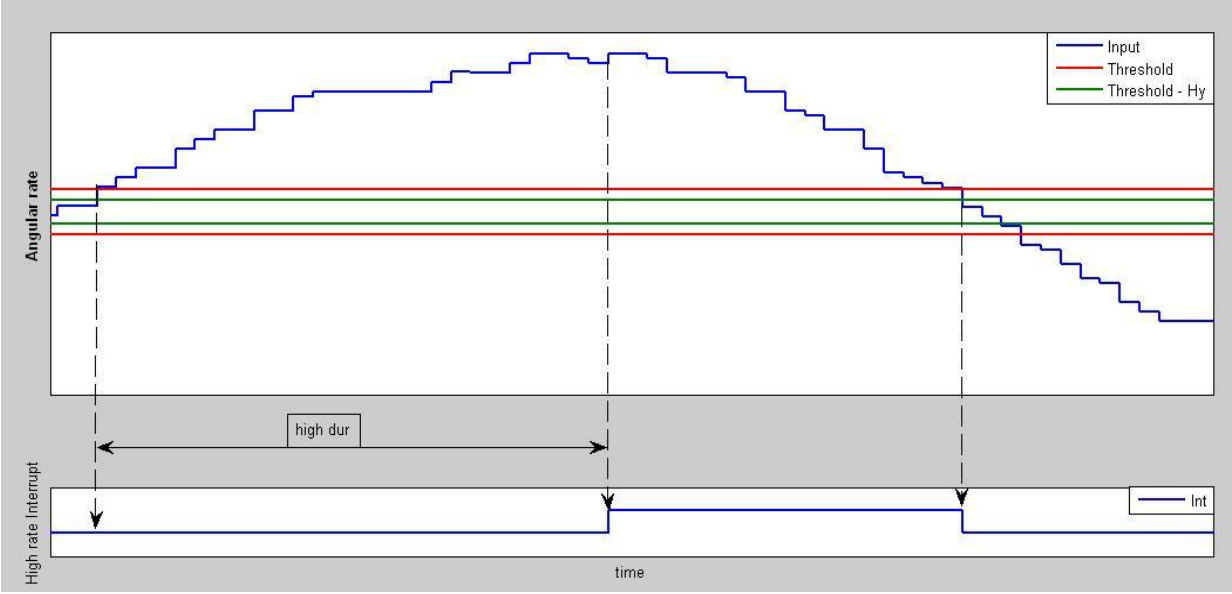


Figure 3: High rate interrupt

The high-rate interrupt is enabled (disabled) per axis by writing ´1´ (´0´) to bits *GYRO_HIGH_RATE* in the *INT_EN* register and for each axis by writing to the *HR_X_AXIS*, *HR_Y_AXIS*, and *HR_Z_AXIS*, respectively in the GYR_INT_SETTING register. The high-rate threshold is set through the *HR_<axis>_*Threshold bits in the appropriate *GYR_HR_<axis>_SET register*. The meaning of an LSB of *HR_<axis>_Threshold* depends on the selected °/s-range: it corresponds to 62.5°/s in 2000°/s-range, 31.25°/s in 1000°/s-range, 15.625°/s in 500°/s -range ...). The *HR_<axis>_Threshold* register setting 0 corresponds to 62.26°/s in 2000°/s-range, 31.13°/s in 1000°/s-range, 15.56°/s in 500°/s-range .... Therefore the maximum value is 1999.76°/s in 2000°/s-range (999.87°/s 1000°/s-range, 499.93°/s in 500°/s -range ...).
A hysteresis can be selected by setting the *HR_<axis>_THRES_HYST* bits. Analogously to threshold, the meaning of an LSB of *HR_<axis>_THRES_HYST* bits is °/s-range dependent: The *HR_<axis>_THRES_HYST* register setting 0 corresponds to an angular rate difference of 62.26°/s in 2000°/s-range, 31.13°/s in 1000°/s-range, 15.56°/s in 500°/s-range .... The meaning of an LSB of *HR_<axis>_THRES_HYST* depends on the selected °/s-range too: it corresponds to 62.5°/s in 2000°/s-range, 31.25°/s in 1000°/s-range, 15.625°/s in 500°/s -range ...).

The high-rate interrupt is generated if the absolute value of the angular rate of at least one of the enabled axes (´or´ relation) is higher than the threshold for at least the time defined by the *GYR_DUR_<axis>* register. The interrupt is reset if the absolute value of the angular rate of all enabled axes (´and´ relation) is lower than the threshold minus the hysteresis. In bit *GYR_HIGH_RATE* in the *INT_STA* the interrupt status is stored. The relation between the content of *GYR_DUR_<axis>* and the actual delay of the interrupt generation is delay [ms] = [*GYR_DUR_<axis>* + 1] * 2.5 ms. Therefore, possible delay times range from 2.5 ms to 640 ms.

Table 3-43: High Rate Interrupt parameters and Axis selection

Params	Value	[Reg Addr]: Register Value
Axis selection	X-axis	[GYR_INT_SETTING]: xxx1xxxb
	Y-axis	[GYR_INT_SETTING]: xx1xxxxb
	Z-axis	[GYR_INT_SETTING]: x1xxxxxb
High Rate Filter settings	Filtered	[GYR_INT_SETTING]: 0xxxxxxb
	Unfiltered	[GYR_INT_SETTING]: 1xxxxxxb
Interrupt Settings X-axis	Threshold	[GYR_HR_X_SET]: bit4 : bit0
	Duration	[GYR_DUR_X]: bit7 : bit0
	Hysteresis	[GYR_HR_X_SET]: bit6 : bit5
Interrupt Settings Y-axis	Threshold	[GYR_HR_Y_SET]: bit4 : bit0
	Duration	[GYR_DUR_Y]: bit7 : bit0
	Hysteresis	[GYR_HR_Y_SET]: bit6 : bit5
Interrupt Settings X-axis	Threshold	[GYR_HR_Z_SET]: bit4 : bit0
	Duration	[GYR_DUR_Z]: bit7 : bit0
	Hysteresis	[GYR_HR_Z_SET]: bit6 : bit5

3.7.2.5 Gyroscope Any Motion Interrupt

Any-motion (slope) detection uses the slope between successive angular rate signals to detect changes in motion. An interrupt is generated when the slope (absolute value of angular rate difference) exceeds a preset threshold. It is cleared as soon as the slope falls below the threshold. The principle is made clear in Figure 4.

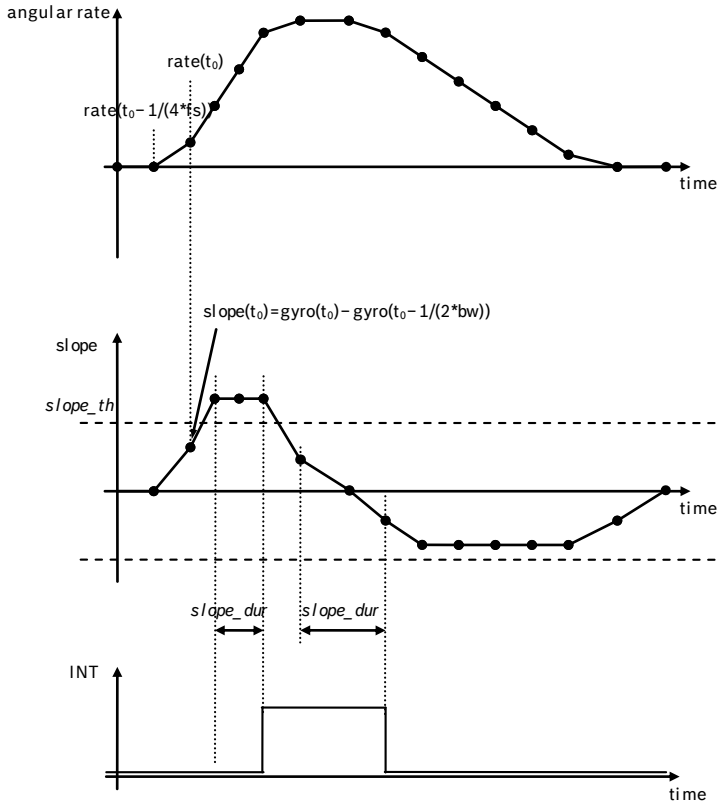


Figure 4: Principle of any-motion detection

The threshold is defined through register GYR_AM_THRES. In terms of scaling 1 LSB of GYR_AM_THRES corresponds to 1 °/s in 2000°/s-range (0.5°/s in 1000°/s-range, 0.25°/s in 500°/s -range ...). Therefore the maximum value is 125°/s in 2000°/s-range (62.5°/s 1000°/s-range, 31.25 in 500°/s -range ...).

The time difference between the successive angular rate signals depends on the selected update rate(fs) which is coupled to the bandwidth and equates to 1/(4*fs) (t=1/(4*fs)). For bandwidth settings with an update rate higher than 400Hz (bandwidth =0,1,2) fs is set to 400Hz.

In order to suppress false triggers, the interrupt is only generated (cleared) if a certain number *N* of consecutive slope data points is larger (smaller) than the slope threshold given by GYR_AM_THRES. This number is set by the Slope Samples bits in the GYR_AM_SET register. It is *N* = [Slope Samples + 1]*4. N is set in samples. Thus the time is scaling with the update rate (fs).

3.7.2.6 Enabling (disabling) for each axis

Any-motion detection can be enabled (disabled) for each axis separately by writing ´1´ (´0´) to bits *AM_X_AXIS, AM_Y_AXIS, AM_Z_AXIS in the GYR_INT_SETTING register*. The criteria for any-motion detection are fulfilled and the Any-Motion interrupt is generated if the slope of any of the enabled axes exceeds the threshold GYR_AM_THRES for [Slope Samples+1]*4 consecutive times. As soon as the slopes of all enabled axes fall or stay below this threshold for [Slope Samples +1]*4 consecutive times the interrupt is cleared unless interrupt signal is latched.

3.7.2.7 Axis of slope / any motion interrupt

The interrupt status is stored in bit *GYRO_AM* in the *INT_EN* register. The Any-motion interrupt supplies additional information about the detected slope.

Table 3-44: Axis selection and any motion interrupt

Params	Value	[Reg Addr]: Register Value
Axis selection	X-axis	[GYR_INT_SETING]: xxxxxx1b
	Y-axis	[GYR_INT_SETING]: xxxxx1xb
	Z-axis	[GYR_INT_SETING]: xxxx1xxb
Any Motion Filter settings	Filtered	[GYR_INT_SETING]: x0xxxxxb
	Unfiltered	[GYR_INT_SETING]: x1xxxxxb
Interrupt Settings	Threshold	[GYR_AM_THRES]: bit6 : bit0
	Slope Samples	[GYR_AM_SET]: bit1 : bit0
	Awake Duration	[GYR_AM_SET]: bit3 : bit2

3.8 Self-Test

3.8.1 Power On Self Test (POST)

During the device startup, a power on self test is executed. This feature checks that the connected sensors and microcontroller are responding / functioning correctly. Following tests are executed

Table 3-45: Power on Self Test

Components	Test type
Accelerometer	Verify chip ID
Magnetometer	Verify chip ID
Gyroscope	Verify chip ID
Microcontroller	Memory Build In Self Test

The results of the POST are stored at register ST_RESULT, where a bit set indicates test passed and cleared indicates self test failed.

3.8.2 Build In Self Test (BIST)

The host can trigger a self test from CONFIG MODE. The test can be triggered by setting bit SELF_TEST in the in the SYS_TRIGGER register, the results are stored in the ST_RESULT register. During the execution of the system test, all other features are paused.

Table 3-46: Power on Self Test

Components	Test type
Accelerometer	built in self test
Magnetometer	built in self test
Gyroscope	built in self test
Microcontroller	No test performed

3.9 Boot loader

The boot loader is located at the start of the program memory and it is executed at each reset / power-on sequence. It first checks the status of the nBOOT_LOAD_PIN.

If the nBOOT_LOAD_PIN is pulled low during reset / power-on sequence, it continues execution in boot loader mode. Otherwise the device continues to boot in application mode.

In case there is a firmware update, then an application note would be available in time with the necessary information to upgrade at the host side. Nevertheless it is recommended that the nBOOT_LOAD_PIN is connected as shown in section 5.

3.10 Calibration

Though the sensor fusion software runs the calibration algorithm of all the three sensors (accelerometer, gyroscope and magnetometer) in the background to remove the offsets, some preliminary steps had to be ensured for this automatic calibration to take place.
The accelerometer and the gyroscope are relatively less susceptible to external disturbances, as a result of which the offset is negligible. Whereas the magnetometer is susceptible to external magnetic field and therefore to ensure proper heading accuracy, the calibration steps described below have to be taken.
Depending on the sensors been selected, the following simple steps had to be taken after every 'Power on Reset' for proper calibration of the device.

3.10.1 **Accelerometer Calibration**

- Place the device in 6 different stable positions for a period of few seconds to allow the accelerometer to calibrate.
- Make sure that there is slow movement between 2 stable positions
- The 6 stable positions could be in any direction, but make sure that the device is lying at least once perpendicular to the x, y and z axis.
- The register CALIB_STAT can be read to see the calibration status of the accelerometer.

3.10.2 **Gyroscope Calibration**

- Place the device in a single stable position for a period of few seconds to allow the gyroscope to calibrate
- The register CALIB_STAT can be read to see the calibration status of the gyroscope.

3.10.3 **Magnetometer Calibration**

Magnetometer in general are susceptible to both hard-iron and soft-iron distortions, but majority of the cases are rather due to the former. And the steps mentioned below are to calibrate the magnetometer for hard-iron distortions.
Nevertheless certain precautions need to be taken into account during the positioning of the sensor in the PCB which is described in our HSMI (Handling, Soldering and Mounting Instructions) application note to avoid unnecessary magnetic influences.

Compass, M4G & NDOF_FMC_OFF:

- Make some random movements (for example: writing the number '8' on air) until the CALIB_STAT register indicates fully calibrated.
- It takes more calibration movements to get the magnetometer calibrated than in the NDOF mode.

NDOF:

- The same random movements have to be made to calibrate the sensor as in the FMC_OFF mode, but here it takes relatively less calibration movements (and slightly higher current consumption) to get the magnetometer calibrated.
- The register CALIB_STAT can be read to see the calibration status of the magnetometer.

3.10.4 **Reuse of Calibration Profile**

Once the device is calibrated, the calibration profile can be reused to get the correct orientation data immediately after 'Power of Reset' (prior to going through the steps mentioned in the above section). However, once the sensor enters the internal calibration routine, the calibration profile is overwritten with the newly obtained sensor offsets and sensor radius. Depending on the application, necessary steps had to be ensured for proper calibration of the sensor.

Reading Calibration profile
The calibration profile includes sensor offsets and sensor radius. Host system can read the offsets and radius only after a full calibration is achieved and the operation mode is switched to CONFIG_MODE. Refer to sensor offsets and sensor radius registers.

Setting Calibration profile
It is important that the correct offsets and corresponding sensor radius are used. Incorrect offsets may result in unreliable orientation data even at calibration accuracy level 3. To set the calibration profile the following steps need to be taken

1. Select the operation mode to CONFIG_MODE
2. Write the corresponding sensor offsets and radius data
3. Change operation mode to fusion mode

4. Register description

4.1 General Remarks

The entire communication with the device is performed by reading from and writing to registers. Registers have a width of 8 bits. There are several registers which are either completely or partially marked as 'reserved'. Any reserved bit is ignored when it is written and no specific value is guaranteed when read. It is recommended not to use registers at all which are completely marked as 'reserved'. Furthermore it is recommended to mask out (logical and with zero) reserved bits of registers which are partially marked as reserved.

Read-Only Registers are marked as shown in Table 4-1: Register Access Coding. Any attempt to write to these registers is ignored.
There are bits within some registers that trigger internal sequences. These bits are configured for write-only access and read as value ´0´.

4.2 Register map

The register map is separated into two logical pages, Page 1 contains sensor specific configuration data and Page 0 contains all other configuration parameters and output data.

At power-on Page 0 is selected, the PAGE_ID register can be used to identify the current selected page and change between page 0 and page 1.

4.2.1 Register map Page 0

Table 4-1: Register Access Coding

read/write	read only	write only	reserved

Table 4-2: Register Map Page 0

Register Address	Register Name	Default Value	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
7F-6B	Reserved	NA								
6A	MAG_RADIUS_MSB		Magnetometer Radius							
69	MAG_RADIUS_LSB		Magnetometer Radius							
68	ACC_RADIUS_MSB		Accelerometer Radius							
67	ACC_RADIUS_LSB		Accelerometer Radius							
66	GYR_OFFSET_Z_MSB	0x00	Gyroscope Offset Z <15:8>							
65	GYR_OFFSET_Z_LSB	0x00	Gyroscope Offset Z <7:0>							
64	GYR_OFFSET_Y_MSB	0x00	Gyroscope Offset Y <15:8>							
63	GYR_OFFSET_Y_LSB	0x00	Gyroscope Offset Y <7:0>							
62	GYR_OFFSET_X_MSB	0x00	Gyroscope Offset X <15:8>							
61	GYR_OFFSET_X_LSB	0x00	Gyroscope Offset X <7:0>							
60	MAG_OFFSET_Z_MSB	0x00	Magnetometer Offset Z <15:8>							
5F	MAG_OFFSET_Z_LSB	0x00	Magnetometer Offset Z <7:0>							
5E	MAG_OFFSET_Y_MSB	0x00	Magnetometer Offset Y <15:8>							
5D	MAG_OFFSET_Y_LSB	0x00	Magnetometer Offset Y <7:0>							
5C	MAG_OFFSET_X_MSB	0x00	Magnetometer Offset X <15:8>							
5B	MAG_OFFSET_X_LSB	0x00	Magnetometer Offset X <7:0>							
5A	ACC_OFFSET_Z_MSB	0x00	Accelerometer Offset Z <15:8>							
59	ACC_OFFSET_Z_LSB	0x00	Accelerometer Offset Z <7:0>							
58	ACC_OFFSET_Y_MSB	0x00	Accelerometer Offset Y <15:8>							
57	ACC_OFFSET_Y_LSB	0x00	Accelerometer Offset Y <7:0>							
56	ACC_OFFSET_X_MSB	0x00	Accelerometer Offset X <15:8>							
55	ACC_OFFSET_X_LSB	0x00	Accelerometer Offset X <7:0>							
43 - 54	Reserved	0x00								

Register Address	Register Name	Default Value	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
42	AXIS_MAP_SIGN	TBD						Remapped X axis sign	Remapped Y axis sign	Remapped Z axis sign
41	AXIS_MAP_CONFIG	TBD			Remapped Z axis value		Remapped Y axis value		Remapped X axis value	
40	TEMP_SOURCE	0x02							TEMP_Source <1:0>	
3F	SYS_TRIGGER	0x00	CLK_SEL	RST_INT	RST_SYS					Self_Test
3E	PWR_MODE	0x00							Power Mode <1:0>	
3D	OPR_MODE	0x1C					Operation Mode <3:0>			
3C	Reserved	0xFF								
3B	UNIT_SEL	0x80	ORI_Android_Windows			TEMP_Unit		EUL_Unit	GYR_Unit	ACC_Unit
3A	SYS_ERR	0x00	System Error Code							
39	SYS_STATUS	0x00	System Status Code							
38	SYS_CLK_STATUS	0x00								ST_MAIN_CLK
37	INT_STA	0x00	ACC_NM	ACC_AM	ACC_HIGH_G		GYR_HIGH_RATE	GYRO_AM		
36	ST_RESULT	0x0F					ST_MCU	ST_GYR	ST_MAG	ST_ACC
35	CALIB_STAT	0x00	SYS Calib Status 0:3		GYR Calib Status 0:3		ACC Calib Status 0:3		MAG Calib Status 0:3	
34	TEMP	0x00	Temperature							
33	GRV_Data_Z_MSB	0x00	Gravity Vector Data Z <15:8>							
32	GRV_Data_Z_LSB	0x00	Gravity Vector Data Z <7:0>							
31	GRV_Data_Y_MSB	0x00	Gravity Vector Data Y <15:8>							
30	GRV_Data_Y_LSB	0x00	Gravity Vector Data Y <7:0>							
2F	GRV_Data_X_MSB	0x00	Gravity Vector Data X <15:8>							
2E	GRV_Data_X_LSB	0x00	Gravity Vector Data X <7:0>							
2D	LIA_Data_Z_MBS	0x00	Linear Acceleration Data Z <15:8>							
2C	LIA_Data_Z_LSB	0x00	Linear Acceleration Data Z <7:0>							
2B	LIA_Data_Y_MBS	0x00	Linear Acceleration Data Y <15:8>							
2A	LIA_Data_Y_LSB	0x00	Linear Acceleration Data Y <7:0>							
29	LIA_Data_X_MBS	0x00	Linear Acceleration Data X <15:8>							
28	LIA_Data_X_LSB	0x00	Linear Acceleration Data X <7:0>							
27	QUA_Data_z_MSB	0x00	Quaternion z Data <15:8>							
26	QUA_Data_z_LSB	0x00	Quaternion z Data <7:0>							
25	QUA_Data_y_MSB	0x00	Quaternion y Data <15:8>							
24	QUA_Data_y_LSB	0x00	Quaternion y Data <7:0>							
23	QUA_Data_x_MSB	0x00	Quaternion x Data <15:8>							
22	QUA_Data_x_LSB	0x00	Quaternion x Data <7:0>							
21	QUA_Data_w_MSB	0x00	Quaternion w Data <15:8>							
20	QUA_Data_w_LSB	0x00	Quaternion w Data <7:0>							
1F	EUL_Pitch_MSB	0x00	Pitch Data <15:8>							
1E	EUL_Pitch_LSB	0x00	Pitch Data <7:0>							

Register Address	Register Name	Default Value	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
1D	EUL_Roll_MSB	0x00	Roll Data <15:8>							
1C	EUL_Roll_LSB	0x00	Roll Data <7:0>							
1B	EUL_Heading_MSB	0x00	Heading Data <15:8>							
1A	EUL_Heading_LSB	0x00	Heading Data <7:0>							
19	GYR_DATA_Z_MSB	0x00	Gyroscope Data Z <15:8>							
18	GYR_DATA_Z_LSB	0x00	Gyroscope Data Z <7:0>							
17	GYR_DATA_Y_MSB	0x00	Gyroscope Data Y <15:8>							
16	GYR_DATA_Y_LSB	0x00	Gyroscope Data Y <7:0>							
15	GYR_DATA_X_MSB	0x00	Gyroscope Data X <15:8>							
14	GYR_DATA_X_LSB	0x00	Gyroscope Data X <7:0>							
13	MAG_DATA_Z_MSB	0x00	Magnetometer Data Z <15:8>							
12	MAG_DATA_Z_LSB	0x00	Magnetometer Data Z <7:0>							
11	MAG_DATA_Y_MSB	0x00	Magnetometer Data Y <15:8>							
10	MAG_DATA_Y_LSB	0x00	Magnetometer Data Y <7:0>							
F	MAG_DATA_X_MSB	0x00	Magnetometer Data X <15:8>							
E	MAG_DATA_X_LSB	0x00	Magnetometer Data X <7:0>							
D	ACC_DATA_Z_MSB	0x00	Acceleration Data Z <15:8>							
C	ACC_DATA_Z_LSB	0x00	Acceleration Data Z <7:0>							
B	ACC_DATA_Y_MSB	0x00	Acceleration Data Y <15:8>							
A	ACC_DATA_Y_LSB	0x00	Acceleration Data Y <7:0>							
9	ACC_DATA_X_MSB	0x00	Acceleration Data X <15:8>							
8	ACC_DATA_X_LSB	0x00	Acceleration Data X <7:0>							
7	Page ID	0x00	Page ID							
6	BL_Rev_ID	NA	Bootloader Version							
5	SW_REV_ID_MSB	0x03[5]	SW Revision ID <15:8>							
4	SW_REV_ID_LSB	0x08[6]	SW Revision ID <7:0>							
3	GYR_ID	0x0F	GYRO chip ID							
2	MAG_ID	0x32	MAG chip ID							
1	ACC_ID	0xFB	ACC chip ID							
0	CHIP_ID	0xA0	BNO055 CHIP ID							

[5] The current software version is 0.3.0.8 and therefore the SW_REV_ID_MSB is 0x03. However the register default value is subject to change with respect to the updated software.

[6] The current software version is 0.3.0.8 and therefore the SW_REV_ID_LSB is 0x08. However the register default value is subject to change with respect to the updated software.

4.2.2 **Register map Page 1**

Table4-3: Register Map Page 1

Register Address	Register Name	Default Value	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
7F-60	Reserved	0x00								
5F - 50	UNIQUE_ID	n.a.	BNO unique ID							
4F - 20	Reserved	0x00								
1F	GYR_AM_SET	0x0A					Awake Duration <1:0>		Slope Samples <1:0>	
1E	GYR_AM_THRES	0x04		Gyro Any Motion Threshold <6:0>						
1D	GYR_DUR_Z	0x19	HR_Z_Duration							
1C	GYR_HR_Z_SET	0x01		HR_Z_THRES_HYST <1:0>		HR_Z_Threshold <4:0>				
1B	GYR_DUR_Y	0x19	HR_Y_Duration							
1A	GYR_HR_Y_SET	0x01		HR_Y_THRES_HYST <1:0>		HR_Y_Threshold <4:0>				
19	GYR_DUR_X	0x19	HR_X_Duration							
18	GYR_HR_X_SET	0x01		HR_X_THRES_HYST <1:0>		HR_X_Threshold <4:0>				
17	GYR_INT_SETING	0x00	HR_FILT	AM_FILT	HR_Z_AXIS	HR_Y_AXIS	HR_X_AXIS	AM_Z_AXIS	AM_Y_AXIS	AM_X_AXIS
16	ACC_NM_SET	0x0B		NO/SLOW Motion Duration <5:0>						SMNM
15	ACC_NM_THRE	0x0A	Accelerometer NO/SLOW motion threshold							
14	ACC_HG_THRES	0xC0	Accelerometer High G Threshold							
13	ACC_HG_DURATION	0x0F	Accelerometer High G Duration							
12	ACC_INT_Settings	0x03	HG_Z_AXIS	HG_Y_AXIS	HG_X_AXIS	AM/NM_Z_AXIS	AM/NM_Y_AXIS	AM/NM_X_AXIS	AM_DUR <1:0>	
11	ACC_AM_THRES	0x14	Accelerometer Any motion threshold							
10	INT_EN	0x00	ACC_NM	ACC_AM	ACC_HIGH_G		GYR_HIGH_RATE	GYRO_AM		
F	INT_MSK	0x00	ACC_NM	ACC_AM	ACC_HIGH_G		GYR_HIGH_RATE	GYRO_AM		
E	Reserved	0x00								
D	GYR_Sleep_Config	0x00			AUTO_SLP_DURATION <2:0>			SLP_DURATION <2:0>		
C	ACC_Sleep_Config	0x00				SLP_DURATION <3:0>				SLP_MODE
B	GYR_Config_1	0x00						GYR_Power_Mode <2:0>		
A	GYR_Config_0	0x38			GYR_Bandwidth <2:0>			GYR_Range <2:0>		
9	MAG_Config	0x6D		MAG_Power_mode <1:0>		MAG_OPR_Mode <1:0>		MAG_Data_output_rate <2:0>		
8	ACC_Config	0x0D	ACC_PWR_Mode <2:0>			ACC_BW <2:0>			ACC_Range <1:0>	
7	Page ID	0x01	Page ID							
6 - 0	Reserved	n.a.								

4.3 Register description (Page 0)

4.3.1 CHIP_ID 0x00

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	1	0	1	0	0	0	0	0
Content	BNO055 CHIP ID							

DATA	bits	Description
BNO055 CHIP ID	**<7:0>**	Chip identification code, read-only fixed value 0xA0

4.3.2 ACC_ID 0x01

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	R	r
Reset	0xFB							
Content	ACC chip ID							

DATA	bits	Description
ACC chip ID	<7:0>	Chip ID of the Accelerometer device, read-only fixed value 0xFB

4.3.3 MAG_ID 0x02

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	R	r
Reset	0x32							
Content	MAG chip ID							

DATA	bits	Description
MAG chip ID	<7:0>	Chip ID of the Magnetometer device, read-only fixed value 0x32

4.3.4 GYR_ID 0x03

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	R	r
Reset	0x0F							
Content	GRYO chip ID							

DATA	bits	Description
GYRO chip ID	<7:0>	Chip ID of the Gyroscope device, read-only fixed value 0x0F

4.3.5 SW_REV_ID_LSB 0x04

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset								
Content	SW Revision ID <7:0>							

DATA	bits	Description
SW Revision ID <7:0>	<7:0>	Lower byte of SW Revision ID, read-only fixed value depending on SW revision programmed on microcontroller

4.3.6 SW_REV_ID_MSB 0x05

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset								
Content	SW Revision ID <15:8>							

DATA	bits	Description
SW Revision ID <15:8>	<7:0>	Upper byte of SW Revision ID, read-only fixed value depending on SW revision programmed on microcontroller

4.3.7 BL_REV_ID 0x06

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset								
Content	Bootloader Version							

DATA	bits	Description
Bootloader Version	<7:0>	Identifies the version of the bootloader in the microcontroller, read-only

4.3.8 PAGE ID 0x07

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset	0	0	0	0	0	0	0	0
Content	Page ID							

DATA	bits	Description
Page ID	<7:0>	Read: Number of currently selected page Write: Change page, 0x00 or 0x01

4.3.9 ACC_DATA_X_LSB 0x08

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Acceleration Data X <7:0>							

DATA	bits	Description
Acceleration Data X <7:0>	<7:0>	Lower byte of X axis Acceleration data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.10 ACC_DATA_X_MSB 0x09

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Acceleration Data X <15:8>							

DATA	bits	Description
Acceleration Data X <15:8>	<7:0>	Upper byte of X axis Acceleration data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.11 ACC_DATA_Y_LSB 0x0A

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Acceleration Data Y <7:0>							

DATA	bits	Description
Acceleration Data Y <7:0>	<7:0>	Lower byte of Y axis Acceleration data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.12 ACC_DATA_Y_MSB 0x0B

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Acceleration Data Y <15:8>							

DATA	bits	Description
Acceleration Data Y <15:8>	<7:0>	Upper byte of Y axis Acceleration data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.13 ACC_DATA_Z_LSB 0x0C

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Acceleration Data Z <7:0>							

DATA	bits	Description
Acceleration Data Z <7:0>	<7:0>	Lower byte of Z axis Acceleration data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.14 ACC_DATA_Z_MSB 0x0D

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Acceleration Data Z <15:8>							

DATA	bits	Description
Acceleration Data Z <15:8>	<7:0>	Upper byte of Z axis Acceleration data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.15 MAG_DATA_X_LSB 0x0E

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Magnetometer Data X <7:0>							

DATA	bits	Description
Magnetometer Data X <7:0>	<7:0>	Lower byte of X axis Magnetometer data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.16 MAG_DATA_X_MSB 0x0F

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Magnetometer Data X <15:8>							

DATA	bits	Description
Magnetometer Data X <15:8>	<7:0>	Upper byte of X axis Magnetometer data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.17 MAG_DATA_Y_LSB 0x10

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Magnetometer Data Y <7:0>							

DATA	bits	Description
Magnetometer Data Y <7:0>	<7:0>	Lower byte of Y axis Magnetometer data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.18 MAG_DATA_Y_MSB 0x11

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Magnetometer Data Y <15:8>							

DATA	bits	Description
Magnetometer Data Y <15:8>	<7:0>	Upper byte of Y axis Magnetometer data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.19 MAG_DATA_Z_LSB 0x12

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Magnetometer Data Z <7:0>							

DATA	bits	Description
Magnetometer Data Z <7:0>	<7:0>	Lower byte of Z axis Magnetometer data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.20 MAG_DATA_Z_MSB 0x13

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Magnetometer Data Z <15:8>							

DATA	bits	Description
Magnetometer Data Z <15:8>	<7:0>	Upper byte of Z axis Magnetometer data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.21 GYR_DATA_X_LSB 0x14

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Gyroscope Data X <7:0>							

DATA	bits	Description
Gyroscope Data X <7:0>	<7:0>	Lower byte of X axis Gyroscope data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.22 GYR_DATA_X_MSB 0x15

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Gyroscope Data X <15:8>							

DATA	bits	Description
Gyroscope Data X <15:8>	<7:0>	Upper byte of X axis Gyroscope data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.23 GYR_DATA_Y_LSB 0x16

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Gyroscope Data Y <7:0>							

DATA	bits	Description
Gyroscope Data Y <7:0>	<7:0>	Lower byte of Y axis Gyroscope data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.24 GYR_DATA_Y_MSB 0x17

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Gyroscope Data Y <15:8>							

DATA	bits	Description
Gyroscope Data Y <15:8>	<7:0>	Upper byte of Y axis Gyroscope data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.25 GYR_DATA_Z_LSB 0x18

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Gyroscope Data Z <7:0>							

DATA	bits	Description
Gyroscope Data Z <7:0>	<7:0>	Lower byte of Z axis Gyroscope data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.26 GYR_DATA_Z_MSB 0x19

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Gyroscope Data Z <15:8>							

DATA	bits	Description
Gyroscope Data Z <15:8>	<7:0>	Upper byte of Z axis Gyroscope data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.27 EUL_DATA_X_LSB 0x1A

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Heading Data <7:0>							

DATA	bits	Description
Heading Data <7:0>	<7:0>	Lower byte of heading data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.28 EUL_DATA_X_MSB 0x1B

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Heading Data <15:8>							

DATA	bits	Description
Heading Data <15:8>	<7:0>	Upper byte of heading data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.29 EUL_DATA_Y_LSB 0x1C

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Roll Data <7:0>							

DATA	bits	Description
Roll Data <7:0>	<7:0>	Lower byte of roll data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.30 EUL_DATA_Y_MSB 0x1D

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Roll Data <15:8>							

DATA	bits	Description
Roll Data <15:8>	<7:0>	Upper byte of Y axis roll data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.31 EUL_DATA_Z_LSB 0x1E

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Pitch Data <7:0>							

DATA	bits	Description
Pitch Data <7:0>	<7:0>	Lower byte of pitch data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.32 EUL_DATA_Z_MSB 0x1F

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Pitch Data <15:8>							

DATA	bits	Description
Pitch Data <15:8>	<7:0>	Upper byte of pitch data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.33 QUA_DATA_W_LSB 0x20

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Quaternion Data W <7:0>							

DATA	bits	Description
Quaternion Data W <7:0>	<7:0>	Lower byte of w axis Quaternion data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.34 QUA_DATA_W_MSB 0x21

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Quaternion Data W <15:8>							

DATA	bits	Description
Quaternion Data W <15:8>	<7:0>	Upper byte of w axis Quaternion data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.35 QUA_DATA_X_LSB 0x22

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Quaternion Data X <7:0>							

DATA	bits	Description
Quaternion Data X <7:0>	<7:0>	Lower byte of X axis Quaternion data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.36 QUA_DATA_X_MSB 0x23

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Quaternion Data X <15:8>							

DATA	bits	Description
Quaternion Data X <15:8>	<7:0>	Upper byte of X axis Quaternion data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.37 QUA_DATA_Y_LSB 0x24

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Quaternion Data Y <7:0>							

DATA	bits	Description
Quaternion Data Y <7:0>	<7:0>	Lower byte of Y axis Quaternion data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.38 QUA_DATA_Y_MSB 0x25

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Quaternion Data Y <15:8>							

DATA	bits	Description
Quaternion Data Y <15:8>	<7:0>	Upper byte of Y axis Quaternion data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.39 QUA_DATA_Z_LSB 0x26

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Quaternion Data Z <7:0>							

DATA	bits	Description
Quaternion Data Z <7:0>	<7:0>	Lower byte of Z axis Quaternion data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.40 QUA_DATA_Z_MSB 0x27

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Quaternion Data Z <15:8>							

DATA	bits	Description
Quaternion Data Z <15:8>	<7:0>	Upper byte of Z axis Quaternion data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.41 LIA_DATA_X_LSB 0x28

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Linear Acceleration Data X <7:0>							

DATA	bits	Description
Linear Acceleration Data X <7:0>	<7:0>	Lower byte of X axis Linear Acceleration data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.42 LIA_DATA_X_MSB 0x29

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Linear Acceleration Data X <15:8>							

DATA	bits	Description
Linear Acceleration Data X <15:8>	<7:0>	Upper byte of X axis Linear Acceleration data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.43 LIA_DATA_Y_LSB 0x2A

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Linear Acceleration Data Y <7:0>							

DATA	bits	Description
Linear Acceleration Data Y <7:0>	<7:0>	Lower byte of Y axis Linear Acceleration data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.44 LIA_DATA_Y_MSB 0x2B

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Linear Acceleration Data Y <15:8>							

DATA	bits	Description
Linear Acceleration Data Y <15:8>	<7:0>	Upper byte of Y axis Linear Acceleration data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.45 LIA_DATA_Z_LSB 0x2C

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Linear Acceleration Data Z <7:0>							

DATA	bits	Description
Linear Acceleration Data Z <7:0>	<7:0>	Lower byte of Z axis Linear Acceleration data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.46 LIA_DATA_Z_MSB 0x2D

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Linear Acceleration Data Z <15:8>							

DATA	bits	Description
Linear Acceleration Data Z <15:8>	<7:0>	Upper byte of Z axis Linear Acceleration data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.47 GRV_DATA_X_LSB 0x2E

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Gravity Vector Data X <7:0>							

DATA	bits	Description
Gravity Vector Data X <7:0>	<7:0>	Lower byte of X axis Gravity Vector data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.48 GRV_DATA_X_MSB 0x2F

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Gravity Vector Data X <15:8>							

DATA	bits	Description
Gravity Vector Data X <15:8>	<7:0>	Upper byte of X axis Gravity Vector data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.49 GRV_DATA_Y_LSB 0x30

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Gravity Vector Data Y <7:0>							

DATA	bits	Description
Gravity Vector Data Y <7:0>	<7:0>	Lower byte of Y axis Gravity Vector data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.50 GRV_DATA_Y_MSB 0x31

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Gravity Vector Data Y <15:8>							

DATA	bits	Description
Gravity Vector Data Y <15:8>	<7:0>	Upper byte of Y axis Gravity Vector data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.51 GRV_DATA_Z_LSB 0x32

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Gravity Vector Data Z <7:0>							

DATA	bits	Description
Gravity Vector Data Z <7:0>	<7:0>	Lower byte of Z axis Gravity Vector data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.52 GRV_DATA_Z_MSB 0x33

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Gravity Vector Data Z <15:8>							

DATA	bits	Description
Gravity Vector Data Z <15:8>	<7:0>	Upper byte of Z axis Gravity Vector data, read only The output units can be selected using the UNIT_SEL register and data output type can be changed by updating the Operation Mode in the OPR_MODE register, see section 3.3

4.3.53 **TEMP 0x34**

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Temperature							

DATA	bits	Description
Temperature	<7:0>	Temperature data, read only The output units can be selected using the UNIT_SEL register and data output source can be selected by updating the TEMP_SOURCE register, see section 3.6.5.8

4.3.54 **CALIB_STAT 0x35**

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	SYS Calib Status <0:1>		GYR Calib Status <0:1>		ACC Calib Status <0:1>		MAG Calib Status <0:1>	

DATA	bits	Description
SYS Calib Status <0:1>	<7:6>	Current system calibration status, depends on status of all sensors, read-only Read: 3 indicates fully calibrated; 0 indicates not calibrated
GYR Calib Status <0:1>	<5:4>	Current calibration status of Gyroscope, read-only Read: 3 indicates fully calibrated; 0 indicates not calibrated
ACC Calib Status <0:1>	<3:2>	Current calibration status of Accelerometer, read-only Read: 3 indicates fully calibrated; 0 indicates not calibrated
MAG Calib Status <0:1>	<1:0>	Current calibration status of Magnetometer, read-only Read: 3 indicates fully calibrated; 0 indicates not calibrated

4.3.55 **ST_RESULT 0x36**

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset					1	1	1	1
Content	Reserved				ST_MCU	ST_GYR	ST_MAG	ST_ACC

DATA	bits	Description
ST_MCU	3	Microcontroller self test result. Read: 1 indicated test passed; 0 indicates test failed
ST_GYR	2	Gyroscope self test result. Read: 1 indicated test passed; 0 indicates test failed
ST_MAG	1	Magnetometer self test result. Read: 1 indicated test passed; 0 indicates test failed
ST_ACC	0	Accelerometer self test result. Read: 1 indicated test passed; 0 indicates test failed

4.3.56 **INT_STA 0x37**

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0		0	0		
Content	ACC_NM	ACC_AM	ACC_HIGH_G	Reserved	GYR_HIGH_RATE	GYRO_AM	Reserved	Reserved

DATA	bits	Description
ACC_NM	7	Status of Accelerometer no motion or slow motion interrupt, read only Read: 1 indicates interrupt triggered; 0 indicates no interrupt triggered
ACC_AM	6	Status of Accelerometer any motion interrupt, read only Read: 1 indicates interrupt triggered; 0 indicates no interrupt triggered
ACC_HIGH_G	5	Status of Accelerometer high-g interrupt, read only Read: 1 indicates interrupt triggered; 0 indicates no interrupt triggered
GYR_HIGH_RATE	3	Status of gyroscope high rate interrupt, read only Read: 1 indicates interrupt triggered; 0 indicates no interrupt triggered
GYRO_AM	2	Status of gyroscope any motion interrupt, read only Read: 1 indicates interrupt triggered; 0 indicates no interrupt triggered

4.3.57 **SYS_CLK_STATUS 0x38**

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	ST_MAIN_CLK

DATA	bits	Description
0	0	Indicates that, it is Free to configure the CLK SRC (External or Internal)
1	0	Indicates that, it is in Configuration state

4.3.58 **SYS_STATUS 0x39**

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	System Status Code							

DATA	bits	Description
System Status Code	<7:0>	Read: 0 System idle, 1 System Error, 2 Initializing peripherals 3 System Initialization 4 Executing selftest, 5 Sensor fusion algorithm running, 6 System running without fusion algorithm

4.3.59 SYS_ERR 0x3A

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset								
Content	System Error Code							

DATA	bits	Description
System Error Code	<7:0>	Read the error status from this register if the SYS_STATUS (0x39) register is SYSTEM ERROR (0x01) Read : 0 No error 1 Peripheral initialization error 2 System initialization error 3 Self test result failed 4 Register map value out of range 5 Register map address out of range 6 Register map write error 7 BNO low power mode not available for selected operation mode 8 Accelerometer power mode not available 9 Fusion algorithm configuration error A Sensor configuration error

4.3.60 UNIT_SEL 0x3B

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset	0			0	0	0	0	0
Content	ORI_Android_Windows	reserved		TEMP_Unit	reserved	EUL_Unit	GYR_Unit	ACC_Unit

DATA	bits	Description
ORI_Android_Windows	7	Read: Current selected orientation mode Write: Select orientation mode 0: Windows orientation 1: Android orientation See section 3.6.2 for more details
TEMP_Unit	5	Read: Current selected temperature units Write: Select temperature units 0: Celsius 1: Fahrenheit See section 3.6.1 for more details
EUL_Unit	3	Read: Current selected Euler units Write: Select Euler units 0: Degrees 1: Radians See section 3.6.1 for more details
GYR_Unit	2	Read: Current selected angular rate units Write: Select angular rate units 0: dps 1: rps See section 3.6.1 for more details
ACC_Unit	1	Read: Current selected acceleration units Write: Select acceleration units 0: m/s^2 1: mg See section 3.6.1 for more details

4.3.61 OPR_MODE 0x3D

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access					r/w	r/w	r/w	r/w
Reset								
Content	Reserved				Operation Mode <3:0>			

DATA	bits	Description
Operation Mode <3:0>	<3:0>	Read: Current selected operation mode Write: Select operation mode See section 3.3 for details

4.3.62 PWR_MODE 0x3E

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access							r/w	r/w
Reset								
Content	Reserved						Power Mode <1:0>	

DATA	bits	Description
Power Mode <1:0>	<1:0>	Read: Current selected power mode Write: Select power mode See section 0 for details

4.3.63 SYS_TRIGGER 0x3F

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	w	w	w					w
Reset	0	0	0					0
Content	CLK_SEL	RST_INT	RST_SYS					Self_Test

DATA	bits	Description
CLK_SEL	7	0: Use internal oscillator 1: Use external oscillator. Set this bit only if external crystal is connected
RST_INT	6	Set to reset all interrupt status bits, and INT output
RST_SYS	5	Set to reset system
Self_Test	0	Set to trigger self test

4.3.64 TEMP_SOURCE 0x40

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access							r/w	r/w
Reset								
Content	Reserved						TEMP_Source <1:0>	

DATA	bits	Description
TEMP_Source <1:0>	<1:0>	See section 3.6.5.8 for details

4.3.65 **AXIS_MAP_CONFIG 0x41**

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access			r/w	r/w	r/w	r/w	r/w	r/w
Reset								
Content	Reserved		Remapped Z axis value		Remapped Y axis value		Remapped X axis value	

DATA	bits	Description
Remapped Z axis value	<5:4>	See section 3.4 for details
Remapped Y axis value	<3:2>	See section 3.4 for details
Remapped X axis value	<1:0>	See section 3.4 for details

4.3.66 **AXIS_MAP_SIGN 0x42**

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access						r/w	r/w	r/w
Reset								
Content	Reserved					Remapped X axis sign	Remapped Y axis sign	Remapped Z axis sign

DATA	bits	Description
Remapped X axis sign	2	See section 3.4 for details
Remapped Y axis sign	1	See section 3.4 for details
Remapped Z axis sign	0	See section 3.4 for details

4.3.67 **ACC_OFFSET_X_LSB 0x55**

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset								
Content	Accelerometer Offset X <7:0>							

DATA	bits	Description
Accelerometer Offset X <7:0>	<7:0>	See section 3.6.4 for details

4.3.68 ACC_OFFSET_X_MSB 0x56

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset								
Content	Accelerometer Offset X <15:8>							

DATA	bits	Description
Accelerometer Offset X <15:8>	<7:0>	See section 3.6.4 for details

4.3.69 ACC_OFFSET_Y_LSB 0x57

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset								
Content	Accelerometer Offset Y <7:0>							

DATA	bits	Description
Accelerometer Offset Y <7:0>	<7:0>	See section 3.6.4 for details

4.3.70 ACC_OFFSET_Y_MSB 0x58

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset								
Content	Accelerometer Offset Y <15:8>							

DATA	bits	Description
Accelerometer Offset Y <15:8>	<7:0>	See section 3.6.4 for details

4.3.71 ACC_OFFSET_Z_LSB 0x59

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset								
Content	Accelerometer Offset Z <7:0>							

DATA	bits	Description
Accelerometer Offset Z <7:0>	<7:0>	See section 3.6.4 for details

4.3.72 ACC_OFFSET_Z_MSB 0x5A

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset								
Content	Accelerometer Offset Z <15:8>							

DATA	bits	Description
Accelerometer Offset Z <15:8>	<7:0>	See section 3.6.4 for details

4.3.73 MAG_OFFSET_X_LSB 0x5B

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset								
Content	Magnetometer Data X <7:0>							

DATA	bits	Description
Magnetometer Offset X <7:0>	<7:0>	See section 3.6.4 for details

4.3.74 MAG_OFFSET_X_MSB 0x56C

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset								
Content	Magnetometer Offset X <15:8>							

DATA	bits	Description
Magnetometer Offset X <15:8>	<7:0>	See section 3.6.4 for details

4.3.75 MAG_OFFSET_Y_LSB 0x5D

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset								
Content	Magnetometer Offset Y <7:0>							

DATA	bits	Description
Magnetometer Offset Y <7:0>	<7:0>	See section 3.6.4 for details

4.3.76 MAG_OFFSET_Y_MSB 0x5E

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset								
Content	Magnetometer Offset Y <15:8>							

DATA	bits	Description
Magnetometer Offset Y <15:8>	<7:0>	See section 3.6.4 for details

4.3.77 MAG_OFFSET_Z_LSB 0x5F

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset								
Content	Magnetometer Offset Z <7:0>							

DATA	bits	Description
Magnetometer Offset Z <7:0>	<7:0>	See section 3.6.4 for details

4.3.78 MAG_OFFSET_Z_MSB 0x60

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset								
Content	Magnetometer Offset Z <15:8>							

DATA	bits	Description
Magnetometer Offset Z <15:8>	<7:0>	See section 3.6.4 for details

4.3.79 GYR_OFFSET_X_LSB 0x61

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset								
Content	Gyroscope Data X <7:0>							

DATA	bits	Description
Gyroscope Offset X <7:0>	<7:0>	See section 3.6.4 for details

4.3.80 GYR_OFFSET_X_MSB 0x62

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset								
Content	Gyroscope Offset X <15:8>							

DATA	bits	Description
Gyroscope Offset X <15:8>	<7:0>	See section 3.6.4 for details

4.3.81 GYR_OFFSET_Y_LSB 0x63

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset								
Content	Gyroscope Offset Y <7:0>							

DATA	bits	Description
Gyroscope Offset Y <7:0>	<7:0>	See section 3.6.4 for details

4.3.82 GYR_OFFSET_Y_MSB 0x64

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset								
Content	Gyroscope Offset Y <15:8>							

DATA	bits	Description
Gyroscope Offset Y <15:8>	<7:0>	See section 3.6.4 for details

4.3.83 GYR_OFFSET_Z_LSB 0x65

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset								
Content	Gyroscope Offset Z <7:0>							

DATA	bits	Description
Gyroscope Offset Z <7:0>	<7:0>	See section 3.6.4 for details

4.3.84 GYR_OFFSET_Z_MSB 0x66

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset								
Content	Gyroscope Offset Z <15:8>							

DATA	bits	Description
Gyroscope Offset Z <15:8>	<7:0>	See section 3.6.4 for details

4.3.85 ACC_RADIUS_LSB 0x67

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset								
Content	Accelerometer Radius <7:0>							

DATA	bits	Description
Gyroscope Offset Z <7:0>	<7:0>	See section 3.6.4for details

4.3.86 ACC_RADIUS_MSB 0x68

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset								
Content	Accelerometer Radius <15:8>							

DATA	bits	Description
Gyroscope Offset Z <15:8>	<7:0>	See section 3.6.4 for details

4.3.87 MAG_RADIUS_LSB 0x69

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset								
Content	Magnetometer Radius <7:0>							

DATA	bits	Description
Gyroscope Offset Z <7:0>	<7:0>	See section 3.6.4 for details

4.3.88 MAG_RADIUS_MSB 0x6A

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset								
Content	Magnetometer Radius <15:8>							

DATA	bits	Description
Gyroscope Offset Z <15:8>	<7:0>	See section 3.6.4 for details

4.4 Register description (Page 1)

4.4.1 Page ID 0x07

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset	0	0	0	0	0	0	0	0
Content	Page ID							

DATA	bits	Description
Page ID	<7:0>	Read: Number of currently selected page Write: Change page, 0x00 or 0x01

4.4.2 ACC_Config 0x08

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset	0	0	0	0	1	1	0	1
Content	ACC_PWR_Mode <2:0>			ACC_BW <2:0>			ACC_Range <1:0>	

DATA	bits	Description
ACC_PWR_Mode <2:0>	<7:5>	Read: current selected power mode Write: can only be changed in sensor mode, see section 3.5.2
ACC_BW <2:0>	<4:3>	Read: current selected bandwidth Write: can only be changed in sensor mode, see section 3.5.2
ACC_Range <1:0>	<2:0>	Read: current selected range Write: can only be changed in sensor mode, see section 3.5.2

4.4.3 MAG_Config 0x09

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset	0	0	0	0	1	0	1	1
Content	reserved	MAG_Power_mode <1:0>		MAG_OPR_Mode <1:0>		MAG_Data_output_rate <2:0>		

DATA	bits	Description
MAG_Power_mode <1:0>	<6:5>	Read: current selected power mode Write: can only be changed in sensor mode, see section 3.5.4
MAG_OPR_Mode <1:0>	<4:3>	Read: current selected operation mode Write: can only be changed in sensor mode, see section 3.5.4
MAG_Data_output_rate <2:0>	<2:0>	Read: current selected data output rate Write: can only be changed in sensor mode, see section 3.5.4

4.4.4 GYR_Config_0 0x0A

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset	0	0	1	1	1	0	0	0
Content	reserved		GYR_Bandwidth <2:0>			GYR_Range <2:0>		

DATA	bits	Description
GYR_Bandwidth <2:0>	<5:3>	Read: current selected bandwidth Write: can only be changed in sensor mode, see section 3.5.3
GYR_Range <2:0>	<2:0>	Read: current selected range Write: can only be changed in sensor mode, see section 3.5.3

4.4.5 GYR_Config_1 0x0B

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset	0	0	0	0	0	0	0	0
Content	reserved					GYR_Power_Mode <2:0>		

DATA	bits	Description
GYR_Power_Mode <2:0>	<2:0>	Read: current selected power mode Write: can only be changed in sensor mode, see section 3.5.3

4.4.6 ACC_Sleep_Config 0x0C

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset								
Content	reserved			SLP_DURATION <3:0>				SLP_MODE

DATA	bits	Description
SLP_DURATION <3:0>	<4:1>	Write: The sleep duration for accelerometer low power mode can be only configured in the sensor operation mode where no fusion library is running. Following sleep phase duration is possible to set.

SLP_DURATION	Accelerometer Sleep Phase Duration
0000b	0.5 ms
0001b	0.5 ms
0010b	0.5 ms
0011b	0.5 ms
0100b	0.5 ms
0101b	0.5 ms
0110b	1 ms
0111b	2 ms
1000b	4 ms
1001b	6 ms
1010b	10 ms
1011b	25 ms
1100b	50 ms
1101b	100 ms
1110b	500 ms
1111b	1 ms

SLP_MODE	0	The sleep timer mode for accelerometer low power mode can be only configured in the sensor operation mode where no fusion library is running Write 0: use event driven time-base mode 1: use equidistant sampling time-base mode

4.4.7 GYR_Sleep_Config 0x0D

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset								
Content	reserved		AUTO_SLP_DURATION <2:0>			SLP_DURATION <2:0>		

DATA	bits	Description
AUTO_SLP_DURATION <2:0>	<5:3>	The Gyroscope can be configures in the advanced power mode to optimize the power consumption. This can be only done if the selected operation mode in sensor mode. The auto sleep duration is the wake up duration of gyroscope during the duty cycling between normal and fast-power up mode. Possible configuration for auto sleep duration are:
SLP_DURATION <2:0>	<2:0>	The Gyroscope can be configures in the advanced power mode to optimize the power consumption. This can be only done if the selected operation mode in sensor mode. The sleep duration is the sleep time of gyroscope during the duty cycling between normal and fast-power up mode. Possible configuration for sleep duration are:

Auto sleep duration	Time (ms)
000b	Not allowed
001b	4 ms
010b	5 ms
011b	8 ms
100b	10 ms
101b	15 ms
110b	20 ms
111b	40 ms

Sleep duration	Time (ms)
000b	2 ms
001b	4 ms
010b	5 ms
011b	8 ms
100b	10 ms
101b	15 ms
110b	18 ms
111b	20 ms

The only restriction for the use of the power save mode comes from the configuration of the digital filter bandwidth of gyroscope. For each bandwidth configuration, minimum auto sleep duration must be ensured. For example, for bandwidth = 47Hz, the minimum auto sleep duration is 5ms. This is specified in the table below. For sleep duration, there is no restriction.

Gyroscope bandwidth (Hz)	Mini Autosleep duration (ms)
32 Hz	20 ms
64 Hz	10 ms
12 Hz	20 ms
23 Hz	10 ms
47 Hz	5 ms
116 Hz	4 ms
230 Hz	4 ms
Unfiltered (523 Hz)	4 ms

4.4.8 INT_MSK 0x0F

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset	0	0	0		0	0		
Content	ACC_NM	ACC_AM	ACC_HIGH_G	reserved	GYR_HIGH_RATE	GYRO_AM	reserved	reserved

DATA	bits	Description
ACC_NM	7	Masking of Accelerometer no motion or slow motion interrupt, when enabled the interrupt will update the INT_STA register and trigger a change on the INT pin, when disabled only the INT_STA register will be updated. Read: 1: Enabled / 0: Disabled Write: 1: Enable / 0: Disable
ACC_AM	6	Masking of Accelerometer any motion interrupt, when enabled the interrupt will update the INT_STA register and trigger a change on the INT pin, when disabled only the INT_STA register will be updated. Read: 1: Enabled / 0: Disabled Write: 1: Enable / 0: Disable
ACC_HIGH_G	5	Masking of Accelerometer high-g interrupt, when enabled the interrupt will update the INT_STA register and trigger a change on the INT pin, when disabled only the INT_STA register will be updated. Read: 1: Enabled / 0: Disabled Write: 1: Enable / 0: Disable
GYR_HIGH_RATE	3	Masking of gyroscope high rate interrupt, when enabled the interrupt will update the INT_STA register and trigger a change on the INT pin, when disabled only the INT_STA register will be updated. Read: 1: Enabled / 0: Disabled Write: 1: Enable / 0: Disable
GYRO_AM	2	Masking of gyroscope any motion interrupt, when enabled the interrupt will update the INT_STA register and trigger a change on the INT pin, when disabled only the INT_STA register will be updated. Read: 1: Enabled / 0: Disabled Write: 1: Enable / 0: Disable

4.4.9 **INT_EN 0x10**

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset	0	0	0		0	0		
Content	ACC_NM	ACC_AM	ACC_HIGH_G	reserved	GYR_HIGH_RATE	GYRO_AM	reserved	reserved

DATA	bits	Description
ACC_NM	7	Status of Accelerometer no motion or slow motion interrupt Read: 1: Enabled / 0: Disabled Write: 1: Enable / 0: Disable interrupt
ACC_AM	6	Status of Accelerometer any motion interrupt Read: 1: Enabled / 0: Disabled Write: 1: Enable / 0: Disable interrupt
ACC_HIGH_G	5	Status of Accelerometer high-g interrupt Read: 1: Enabled / 0: Disabled Write: 1: Enable / 0: Disable interrupt
GYR_HIGH_RATE	3	Status of gyroscope high rate interrupt Read: 1: Enabled / 0: Disabled Write: 1: Enable / 0: Disable interrupt
GYRO_AM	2	Status of gyroscope any motion interrupt Read: 1: Enabled / 0: Disabled Write: 1: Enable / 0: Disable interrupt

4.4.10 **ACC_AM_THRES 0x11**

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset	0	0	0	1	0	1	0	0
Content	Accelerometer Any motion threshold							

DATA	bits	Description
Accelerometer Any motion threshold	<7:0>	Threshold used for the any-motion interrupt. The threshold value is dependent on the accelerometer range selected in the ACC_Config register. 1 LSB = 3.91 mg (2-g range) 1 LSB = 7.81 mg (4-g range) 1 LSB = 15.63 mg (8-g range) 1 LSB = 31.25 mg (16-g range)

4.4.11 ACC_INT_Settings 0x12

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset	0	0	0	0	0	0	1	1
Content	HG_Z_AXIS	HG_Y_AXIS	HG_X_AXIS	AM/NM_Z_AXIS	AM/NM_Y_AXIS	AM/NM_X_AXIS	AM_DUR <1:0>	

DATA	bits	Description
HG_Z_AXIS	7	Select which axis of the accelerometer is used to trigger a high-G interrupt 1: Enabled; 0: Disabled
HG_Y_AXIS	6	Select which axis of the accelerometer is used to trigger a high-G interrupt 1: Enabled; 0: Disabled
HG_X_AXIS	5	Select which axis of the accelerometer is used to trigger a high-G interrupt 1: Enabled; 0: Disabled
AM/NM_Z_AXIS	4	Select which axis of the accelerometer is used to trigger a any motion or no motion interrupt 1: Enabled; 0: Disabled
AM/NM_Y_AXIS	3	Select which axis of the accelerometer is used to trigger a any motion or no motion interrupt 1: Enabled; 0: Disabled
AM/NM_X_AXIS	2	Select which axis of the accelerometer is used to trigger a any motion or no motion interrupt 1: Enabled; 0: Disabled
AM_DUR <1:0>	<1:0>	Any motion interrupt triggers if [AM_DUR<1:0>+1] consecutive data points are above the any motion interrupt threshold define in ACC_AM_THRES register

4.4.12 ACC_HG_DURATION 0x13

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset	0	0	0	0	1	1	1	1
Content	Accelerometer High G Duration							

DATA	bits	Description
Accelerometer High G Duration	<7:0>	The high-g interrupt trigger delay according to [ACC_HG_DURATION + 1] * 2 ms in a range from 2 ms to 512 ms;

4.4.13 ACC_HG_THRES 0x14

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset	1	1	0	0	0	0	0	0
Content	Accelerometer High G Threshold							

DATA	bits	Description
Accelerometer High G Threshold	<7:0>	Threshold used high-g interrupt. The threshold value is dependent on the accelerometer range selected in the ACC_Config register. 1 LSB = 7.81 mg (2-g range 1 LSB = 15.63 mg (4-g range) 1 LSB = 31.25 mg (8-g range) 1 LSB = 62.5 mg (16-g range)

4.4.14 ACC_NM_THRES 0x15

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset	0	0	0	0	1	0	1	0
Content	Accelerometer NO/SLOW motion threshold							

DATA	bits	Description
Accelerometer NO/SLOW motion threshold	<7:0>	Threshold used for the Slow motion or no motion interrupt. The threshold value is dependent on the accelerometer range selected in the ACC_Config register. 1 LSB = 3.91 mg (2-g range) 1 LSB = 7.81 mg (4-g range) 1 LSB = 15.63 mg (8-g range) 1 LSB = 31.25 mg (16-g range)

4.4.15 ACC_NM_SET 0x16

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset		0	0	0	1	0	1	1
Content	reserved	slo_no_mot_dur <5:0>						SMNM

DATA	bits	Description
slo_no_mot_dur <5:0>	<6:1>	Function depends on whether the slow-motion or no-motion interrupt function has been selected. If the slow-motion interrupt function has been enabled (SMNM = '0') then [slo_no_mot_dur<1:0>+1] consecutive slope data points must be above the slow/no-motion threshold (ACC_NM_THRES) for the slow-/no-motion interrupt to trigger. If the no-motion interrupt function has been enabled (SMNM = '1') then slo_no_motion_dur<5:0> defines the time for which no slope data points must exceed the slow/no-motion threshold (ACC_NM_THRES) for the slow/no-motion interrupt to trigger. The delay time in seconds may be calculated according with the following equation: slo_no_mot_dur<5:4>='b00' [slo_no_mot_dur<3:0> + 1] slo_no_mot_dur<5:4>='b01' [slo_no_mot_dur<3:0> * 4 + 20] slo_no_mot_dur<5>='1' [slo_no_mot_dur<4:0> * 8 + 88]
SMNM	0	Select slow motion or no motion interrupt 1: Slow motion; 0: No motion

4.4.16 GYR_INT_SETTING 0x17

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset	0	0	0	0	0	0	0	0
Content	HR_FILT	AM_FILT	HR_Z_AXIS	HR_Y_AXIS	HR_X_AXIS	AM_Z_AXIS	AM_Y_AXIS	AM_X_AXIS

DATA	bits	Description
HR_FILT	7	'1' ('0') selects unfiltered (filtered) data for high rate interrupt
AM_FILT	6	'1' ('0') selects unfiltered (filtered) data for any motion interrupt
HR_Z_AXIS	5	1' ('0') enables (disables) high rate interrupt for z-axis
HR_Y_AXIS	4	1' ('0') enables (disables)) high rate interrupt for y-axis
HR_X_AXIS	3	1' ('0') enables (disables)) high rate interrupt for x-axis
AM_Z_AXIS	2	1' ('0') enables (disables) any motion interrupt for z-axis
AM_Y_AXIS	1	1' ('0') enables (disables) any motion interrupt for y-axis
AM_X_AXIS	0	1' ('0') enables (disables) any motion interrupt for x-axis

4.4.17 GYR_HR_X_SET 0x18

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset	0	0	0	0	0	0	0	1
Content	reserved	HR_X_THRES_HYST <1:0>		HR_X_Threshold <4:0>				

DATA	bits	Description
HR_X_THRES_HYST <1:0>	<6:5>	High rate hysteresis for X axis = (255 + 256 * HR_X_THRES_HYST) *4 LSB The high rate value scales with the range setting 1 LSB = 62.26°/s in 2000°/s-range 1 LSB = 31.13°/s in 1000°/s-range 1 LSB = 15.56°/s in 500°/s -range …
HR_X_Threshold <4:0>	<4:0>	High rate threshold is for the gyroscope X axis. The threshold value is dependent on the gyroscope range selected in the GRY_Config_0 register. 1 LSB = 62.5°/s in 2000°/s-range 1 LSB = 31.25°/s in 1000°/s-range 1 LSB = 15.625°/s in 500°/s -range …

4.4.18 GYR_DUR_X 0x19

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset	0	0	0	1	1	0	0	1
Content	HR_X_Duration							

DATA	bits	Description
HR_X_Duration	<7:0>	High rate duration = (1 + HR_X_Duration)*2.5ms

4.4.19 GYR_HR_Y_SET 0x1A

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	1
Content	reserved	HR_Y_THRES_HYST <1:0>		HR_Y_Threshold <4:0>				

DATA	bits	Description
HR_Y_THRES_HYST <1:0>	<6:5>	High rate hysteresis for Y axis = (255 + 256 * HR_Y_THRES_HYST) *4 LSB The high rate value scales with the range setting 1 LSB = 62.26°/s in 2000°/s-range 1 LSB = 31.13°/s in 1000°/s-range 1 LSB = 15.56°/s in 500°/s -range …
HR_Y_Threshold <4:0>	<4:0>	High rate threshold is for the gyroscope Y axis. The threshold value is dependent on the gyroscope range selected in the GRY_Config_0 register. 1 LSB = 62.5°/s in 2000°/s-range 1 LSB = 31.25°/s in 1000°/s-range 1 LSB = 15.625°/s in 500°/s -range …

4.4.20 GYR_DUR_Y 0x1B

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset	0	0	0	1	1	0	0	1
Content	HR_Y_Duration							

DATA	bits	Description
HR_Y_Duration	<7:0>	High rate duration = (1 + HR_Y_Duration)*2.5ms

4.4.21 GYR_HR_Z_SET 0x1C

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	1
Content	reserved	HR_Z_THRES_HYST <1:0>		HR_Z_Threshold <4:0>				

DATA	bits	Description
HR_Z_THRES_HYST <1:0>	<6:5>	High rate hysteresis for Z axis = (255 + 256 * HR_Z_THRES_HYST) *4 LSB The high rate value scales with the range setting 1 LSB = 62.26°/s in 2000°/s-range 1 LSB = 31.13°/s in 1000°/s-range 1 LSB = 15.56°/s in 500°/s -range …
HR_Z_Threshold <4:0>	<4:0>	High rate threshold is for the gyroscope Z axis. The threshold value is dependent on the gyroscope range selected in the GRY_Config_0 register. 1 LSB = 62.5°/s in 2000°/s-range 1 LSB = 31.25°/s in 1000°/s-range 1 LSB = 15.625°/s in 500°/s -range …

4.4.22 GYR_DUR_Z 0x1D

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset	0	0	0	1	1	0	0	1
Content	HR_Z_Duration							

DATA	bits	Description
HR_Z_Duration	<7:0>	High rate duration = (1 + HR_Z_Duration)*2.5ms

4.4.23 GYR_AM_THRES 0x1E

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset	0	0	0	0	0	1	0	0
Content	reserved	Gyro Any Motion Threshold <6:0>						

DATA	bits	Description
Gyro Any Motion Threshold <6:0>	<6:0>	Any motion threshold is for the gyroscope any motion interrupt. The threshold value is dependent on the gyroscope range selected in the GRY_Config_0 register. 1 LSB = 1 °/s in 2000°/s-range 1 LSB = 0.5°/s in 1000°/s-range 1 LSB = 0.25°/s in 500°/s -range …

4.4.24 GYR_AM_SET 0x1F

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
Reset	0	0	0	0	1	0	1	0
Content	reserved				Awake Duration <1:0>		Slope Samples <1:0>	

DATA	bits	Description
Awake Duration <1:0>	<3:2>	0=8 samples, 1=16 samples, 2=32 samples, 3=64 samples
Slope Samples <1:0>	<1:0>	Any motion interrupt triggers if [Slope Samples + 1]*4 consecutive data points are above the any motion interrupt threshold define in GYRO_AM_THRES register

4.5 Digital Interface

The BNO055 supports two digital interfaces for communication between the salve and host device: I^2C which supports the HID-I2C protocol and I2C Standard and Fast modes; and the UART interface.

The active interface is selected by the state of the protocol select pins (PS1 and PS0), Table 4-4 shows the mapping between the protocol select pins and the selected interface mode.

Table 4-4: protocol select pin mapping

PS1	PS0	Functionality
0	0	Standard/Fast I2C Interface
0	1	HID over I2C
1	0	UART Interface
1	1	Reserved

It is not allowed to keep the protocol select pins floating.

Both digital interfaces share partially the same pins, the pin mapping for each interface is shown in Table 4-5.

Table 4-5: Mapping of digital interface pins

PIN	I2C Interfaces (PS1=0b0)	UART Interface (PS1.PS0=0b10)
COM0	SDA	Tx
COM1	SCL	Rx
COM2	GNDIO	
COM3	I2C address select	

The following table shows the electrical specifications of the interface pins:

Table 4-6: Electrical specification of the interface pins

Parameter	Symbol	Condition	Min	Typ	Max	Units
Pull-up Resistance, COM3 pin	R_{up}	Internal Pull-up Resistance to VDDIO	20	40	60	kΩ
Input Capacitance	C_{in}			5	10	pF
I²C Bus Load Capacitance (max. drive capability)	C_{I2C_Load}				400	pF

4.6 I2C Protocol

The I²C bus uses SCL (= SCx pin, serial clock) and SDA (= SDx pin, serial data input and output) signal lines. Both lines are connected to V_{DDIO} externally via pull-up resistors so that they are pulled high when the bus is free.

The I²C interface of the BNO055 is compatible with the I²C Specification UM10204 Rev. 03 (19 June 2007), available at http://www.nxp.com. The BNO055 supports I²C standard mode and fast mode, only 7-bit address mode is supported. The BNO055 I²C interface uses clock stretching.

The default I²C address of the BNO055 device is 0101001b (0x29). The alternative address 0101000b (0x28), in I2C mode the input pin COM3 can be used to select between the primary and alternative I2C address as shown in Table 4-7.

Table 4-7: I2C address selection

I2C configuration	COM3_state	I2C address
Slave	HIGH	0x29
Slave	LOW	0x28
HID-I2C	X	0x40

The timing specification for I²C of the BNO055 is given in Table 4-8: I²C timings:

Table 4-8: I²C timings

Parameter	Symbol	Condition	Min	Max	Units
Clock Frequency	f_{SCL}			400	kHz
SCL Low Period	t_{LOW}		1.3		µs
SCL High Period	t_{HIGH}		0.6		
SDA Setup Time	t_{SUDAT}		0.1		
SDA Hold Time	t_{HDDAT}		0.0		
Setup Time for a repeated Start Condition	t_{SUSTA}		0.6		
Hold Time for a Start Condition	t_{HDSTA}		0.6		
Setup Time for a Stop Condition	t_{SUSTO}		0.6		
Time before a new Transmission can start	t_{BUF}		1.3		
Idle time between write accesses, normal mode, standby mode, low-power mode 2	$t_{IDLE\_wacc\_nm}$		2		µs
Idle time between write accesses, suspend mode, low-power mode 1	$t_{IDLE\ wacc\ sum}$		450		µs

Figure 5: I²C timing diagram shows the definition of the I²C timings given in Table 4-8:

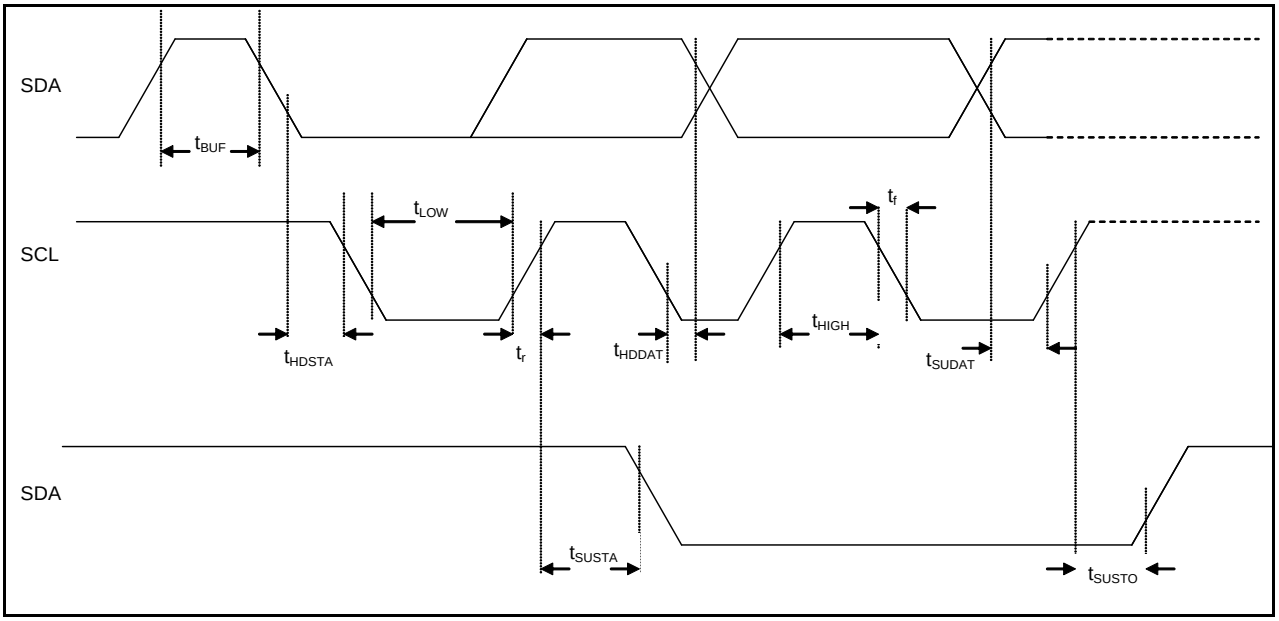


Figure 5: I²C timing diagram

The I²C protocol works as follows:

START: Data transmission on the bus begins with a high to low transition on the SDA line while SCL is held high (start condition (S) indicated by I²C bus master). Once the START signal is transferred by the master, the bus is considered busy.

STOP: Each data transfer should be terminated by a Stop signal (P) generated by master. The STOP condition is a low to HIGH transition on SDA line while SCL is held high.

ACK: Each byte of data transferred must be acknowledged. It is indicated by an acknowledge bit sent by the receiver. The transmitter must release the SDA line (no pull down) during the acknowledge pulse while the receiver must then pull the SDA line low so that it remains stable low during the high period of the acknowledge clock cycle.

In the following diagrams these abbreviations are used:

S	Start
P	Stop
ACKS	Acknowledge by slave
ACKM	Acknowledge by master
NACKM	Not acknowledge by master
RW	Read / Write

A START immediately followed by a STOP (without SCL toggling from ´VDDIO´ to ´GND´) is not supported. If such a combination occurs, the STOP is not recognized by the device.

I²C write access:
I²C write access can be used to write a data byte in one sequence. The sequence begins

with start condition generated by the master, followed by 7 bits slave address and a write bit (RW = 0). The slave sends an acknowledge bit (ACK = 0) and releases the bus. Then the master sends the one byte register address. The slave again acknowledges the transmission and waits for the 8 bits of data which shall be written to the specified register address. After the slave acknowledges the data byte, the master generates a stop signal and terminates the writing protocol.

Example of an I²C write access to the BNO055 (i2c address in this case: 0101000b = 0x28):

Start	Slave address							RW	ACKS	dummy	Register address (0x00 .. 0x7F)								ACKS	Data								ACKS	Stop
S	0	1	0	1	0	0	0	0	A	x	x	x	x	x	x	x	x	A	x	x	x	x	x	x	x	x	A	P

Figure 6: I²C write

I²C read access:

I²C read access also can be used to read one or multiple data bytes in one sequence. A read sequence consists of a one-byte I²C write phase followed by the I²C read phase. The two parts of the transmission must be separated by a repeated start condition (Sr). The I²C write phase addresses the slave and sends the register address to be read. After slave acknowledges the transmission, the master generates again a start condition and sends the slave address together with a read bit (RW = 1). Then the master releases the bus and waits for the data bytes to be read out from slave. After each data byte the master has to generate an acknowledge bit (ACK = 0) to enable further data transfer. A NACKM (ACK = 1) from the master stops the data being transferred from the slave. The slave releases the bus so that the master can generate a STOP condition and terminate the transmission.

The register address is automatically incremented and, therefore, more than one byte can be sequentially read out. Once a new data read transmission starts, the start address will be set to the register address specified in the latest I²C write command. By default the start address is set at 0x00. In this way repetitive multi-bytes reads from the same starting address are possible.

Example of an I²C read access to the BNO055:

Start	Slave address							RW	ACKS	dummy	Register address (0x08)								ACKS
S	0	1	0	1	0	0	0	0	A	x	0	0	0	1	0	0	0	A

Start	Slave address							RW	ACKS	Read data (0x08)								ACKM	Read data (0x09)								ACKM	
Sr	0	1	0	1	0	0	0	1	A	x	x	x	x	x	x	x	x	A	x	x	x	x	x	x	x	x	A	«

«	ACKS	Read data (0x0A)								ACKM	Read data (0x0B)								ACKM	
«	A	x	x	x	x	x	x	x	x	A	x	x	x	x	x	x	x	x	A	«

«	ACKS	Read data (0x0C)								ACKM	Read data (0x0D)								NACKM	Stop
«	A	x	x	x	x	x	x	x	x	A	x	x	x	x	x	x	x	x	NA	P

Figure 7: I²C multiple read

4.7 UART Protocol

The BNO055 supports UART interface with the following settings: 115200 bps, 8N1 (8 data bits, no parity bit, one stop bit). The maximum length support for read and write is 128 Byte. The packet structure for register read and write are described below.

Register write

Command:

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	.....	Byte (n+4)
Start Byte	Write	Reg addr	Length	Data 1	…..	Data n
0xAA	0x00	<..>	<..>	<..>	…..	<..>

Acknowledge Response:

Byte 1	Byte 2
Response Header	Status
0xEE	0x01: WRITE_SUCCESS 0x03: WRITE_FAIL 0x04: REGMAP_INVALID_ADDRESS 0x05: REGMAP_WRITE_DISABLED 0x06: WRONG_START_BYTE 0x07: BUS_OVER_RUN_ERROR 0X08: MAX_LENGTH_ERROR 0x09: MIN_LENGTH_ERROR 0x0A: RECEIVE_CHARACTER_TIMEOUT

Register read

Command:

Byte 1	Byte 2	Byte 2	Byte 3
Start Byte	Read	Reg addr	Length
0xAA	0x01	<..>	<..>

Read Success Response:

Byte 1	Byte 2	Byte 3	.....	Byte (n+2)
ResponseByte	length	Data 1	…..	Data n
0xBB	<..>			

Read Failure or Acknowledge Response:

Byte 1	Byte 2
Response Header	Status
0xEE	0x02: READ_FAIL 0x04: REGMAP_INVALID_ADDRESS 0x05: REGMAP_WRITE_DISABLED 0x06: WRONG_START_BYTE 0x07: BUS_OVER_RUN_ERROR 0X08: MAX_LENGTH_ERROR 0x09: MIN_LENGTH_ERROR 0x0A: RECEIVE_CHARACTER_TIMEOUT

4.8 HID over I2C

HID over I2C is a standard interface protocol to connect devices with hosts via I2C. The main advantage of HID is that there exist generic drivers for different input devices (such as sensors) which can be used with sensors that implement the corresponding well defined HID profiles. HID over I2C describes how messages (reports and events) are exchanged between the device and the host. A descriptor of the structure of these reports is provided by the device and read by the host during initialization of the device at host system start.
For detailed information on HID please refer to the HID over I2C documentation from Microsoft.

5. Pin-out and connection diagram

5.1 Pin-out

The pin-out of the LGA package is shown in Figure 8 and the pin function is described in Table 5-1.

Figure 8: Pin-out bottom view

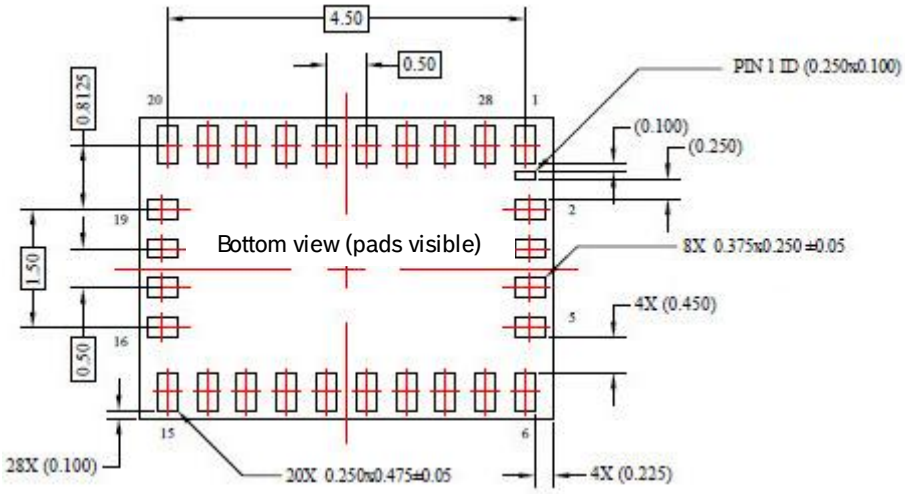
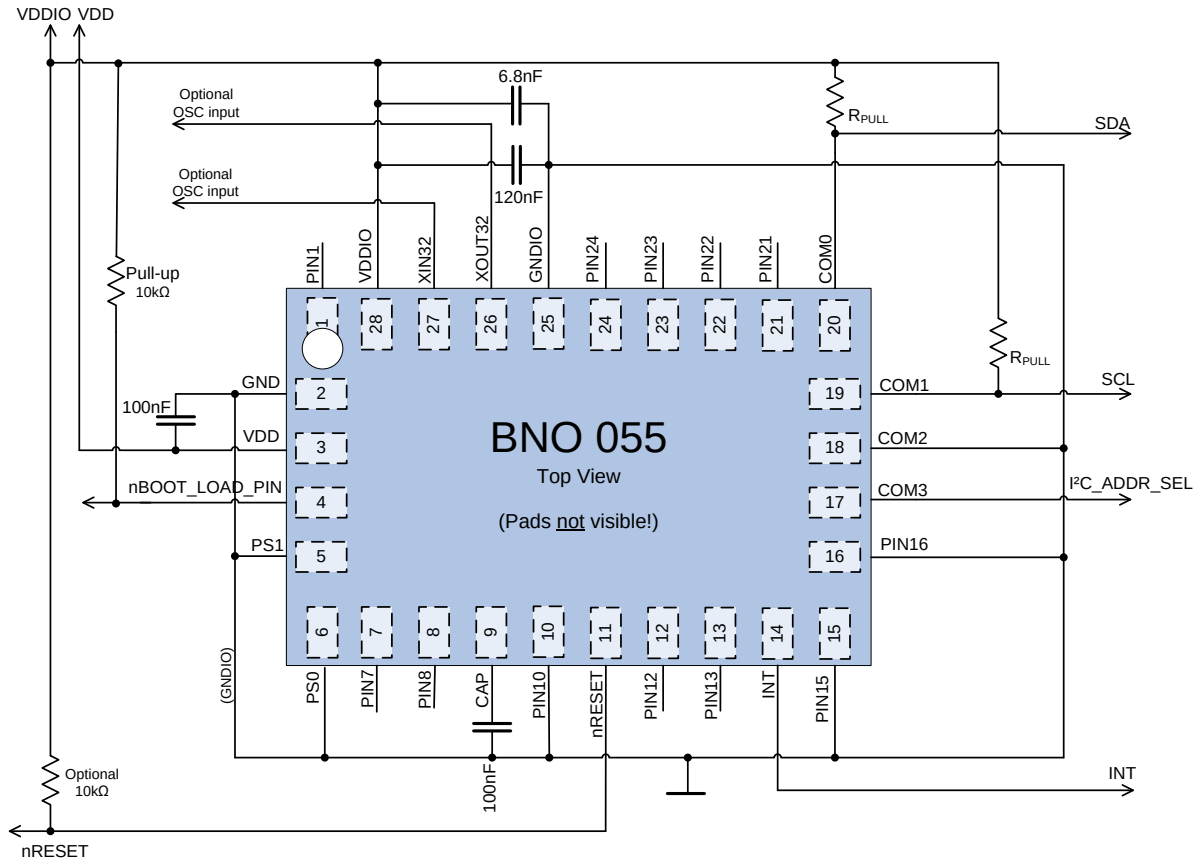


Table 5-1: Pin description

Pin #	Name	I/O Type	Description	Function		
				I2C	UART	HID-I2C
1	PIN1	--	Do not connect	DNC		
2	GND	Ground	GND	GND		
3	VDD	Supply	VDD	VDD		
4	nBOOT_LOAD_PIN	Digital I/O	Bootloader mode select pin (active low)	nBOOT_LOAD_PIN		
5	PS1	Digital in	Protocol select pin 1	GNDIO	VDDIO	GNDIO
6	PS0	Digital in	Protocol select pin 2	GNDIO	GNDIO	VDDIO
7	PIN7	--	Do not connect	DNC		
8	PIN8	--	Do not connect	DNC		
9	CAP	--	External capacitor	CAP		
10	PIN10	Ground	connect to GNDIO	GNDIO		
11	nRESET	--	Reset pin (active low)	nRESET		
12	PIN12	--	Do not connect	DNC		
13	PIN13	--	Do not connect	DNC		
14	INT	Digital Out	Interrupt output	Interrupt		
15	PIN15	Ground	Connect to GNDIO	GNDIO		
16	PIN16	Ground	Connect to GNDIO	GNDIO		
17	COM3	Digital I/O	Digital interface pin 3	I2C address select	GNDIO	GNDIO
18	COM2	Digital I/O	Digital interface pin 2	GNDIO		
19	COM1	Digital I/O	Digital interface pin 1	SCL	Rx	SCL
20	COM0	Digital I/O	Digital interface pin 0	SDA	Tx	SDA
21	PIN21	--	Do not connect	DNC		
22	PIN22	--	Do not connect	DNC		
23	PIN23	--	Do not connect	DNC		
24	PIN24	--	Do not connect	DNC		
25	GNDIO	Ground	GNDIO	GNDIO		
26	XOUT32	Digital Out	Optional OSC port	OSC Output		
27	XIN32	Digital In	Optional OSC port	OSC Input		
28	VDDIO	Supply	VDDIO	VDDIO		

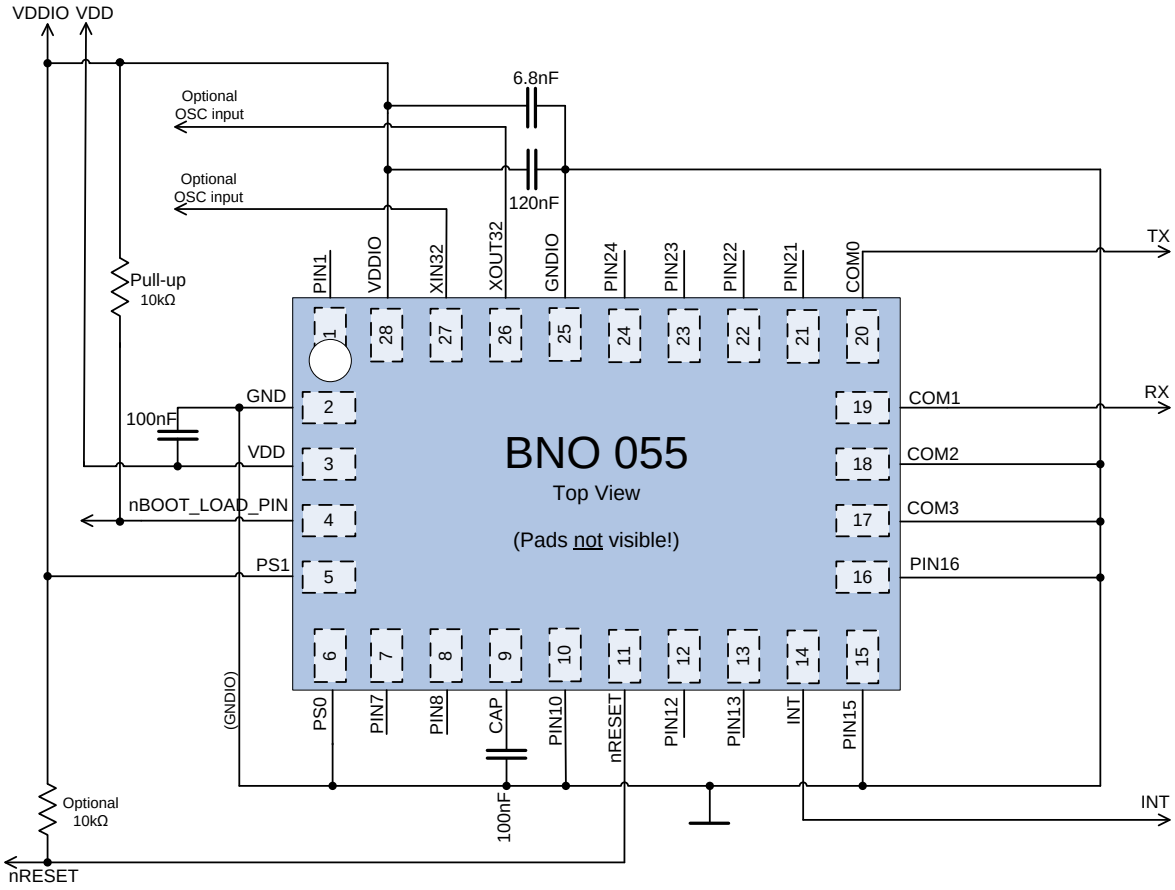
5.2 Connection diagram I²C

Figure 9: I²C connection diagram



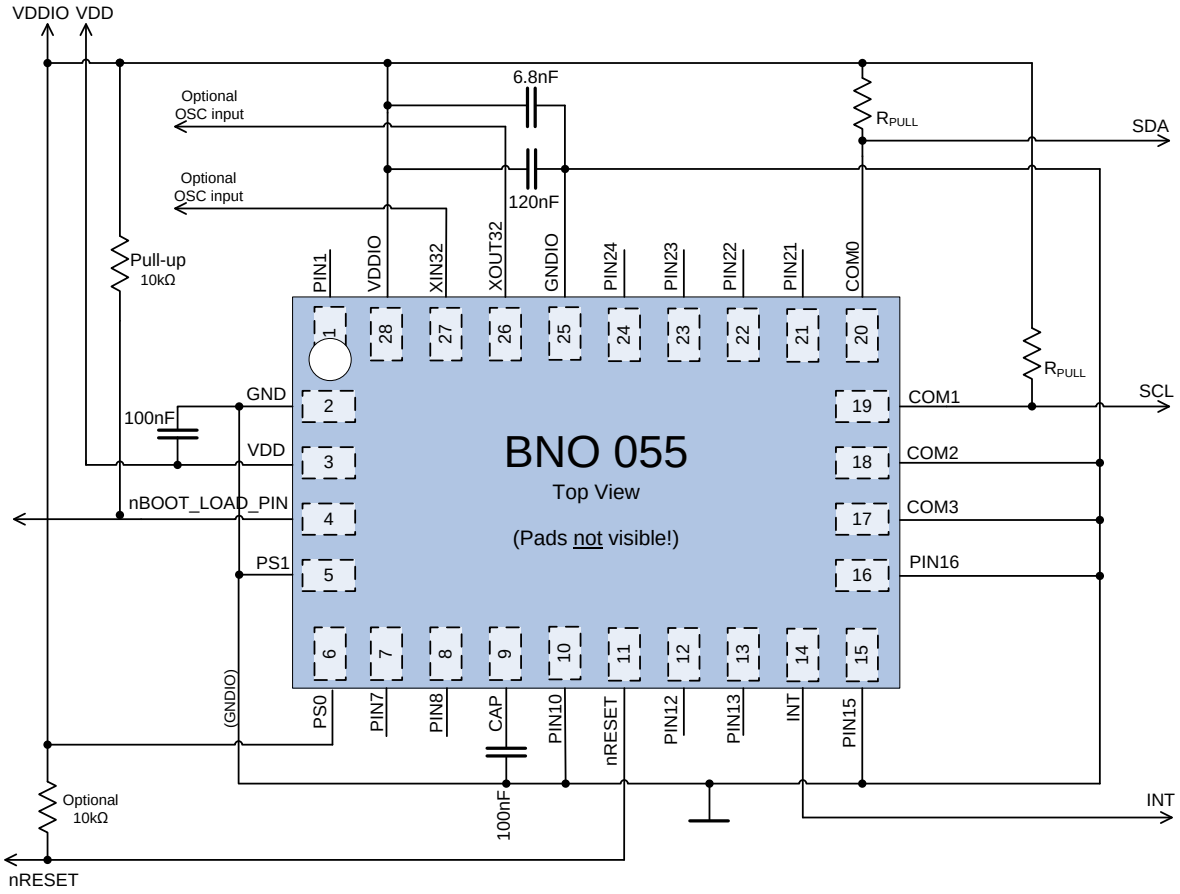
5.3 Connection diagram UART

Figure 10: UART connection diagram



5.4 Connection diagram HID-I2C

Figure 11 : HID via IC connection diagram



5.5 XOUT32 & XIN32 Connections

The BNO055 can run from an internal or external 32 KHz clock source. By default, the internal clock is selected.
An External clock can be selected by setting bit CLK_SEL in the SYSTEM_TRIGGER register. An external 32 KHz crystal oscillator has to be connected to the pins XIN32 and XOUT32 as shown below.

To get the best performance out of BNO055, it is recommended to use the external crystal.

5.5.1 External 32kHz Crystal Oscillator

Figure 12 : External 32kHz Crystal Oscillator with Load Capacitor

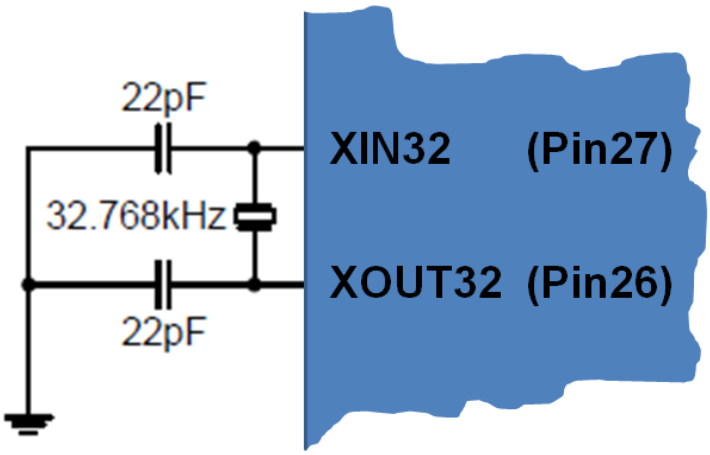


Table 5-2: Crystal Oscillator Source Connections

Pin Name	Recommended Pin Connection	Description
XIN32	Load capacitor 22pF[7][8]	Timer oscillator input
XOUT32	Load capacitor 22pF[7][8]	Timer oscillator output

5.5.2 Internal clock mode

The internal clock can be selected by clearing bit CLK_SEL in the SYSTEM_TRIGGER register. When an internal clock is used, both pins XIN32 and XOUT32 can be left open. The internal clock of the BNO055 can have clock deviation up to ±3%

[7] These values are given only as typical example.
[8] Decoupling capacitor should be placed close to the device for each supply pin pair in the signal group.

6. Package

6.1 Outline dimensions

The sensor package is a standard LGA package; dimensions are shown in the following diagram. Units are in mm. Note: Unless otherwise specified tolerance = decimal ±0.1mm.
The chapter 3.5 provides information regarding the sensor axis orientation.

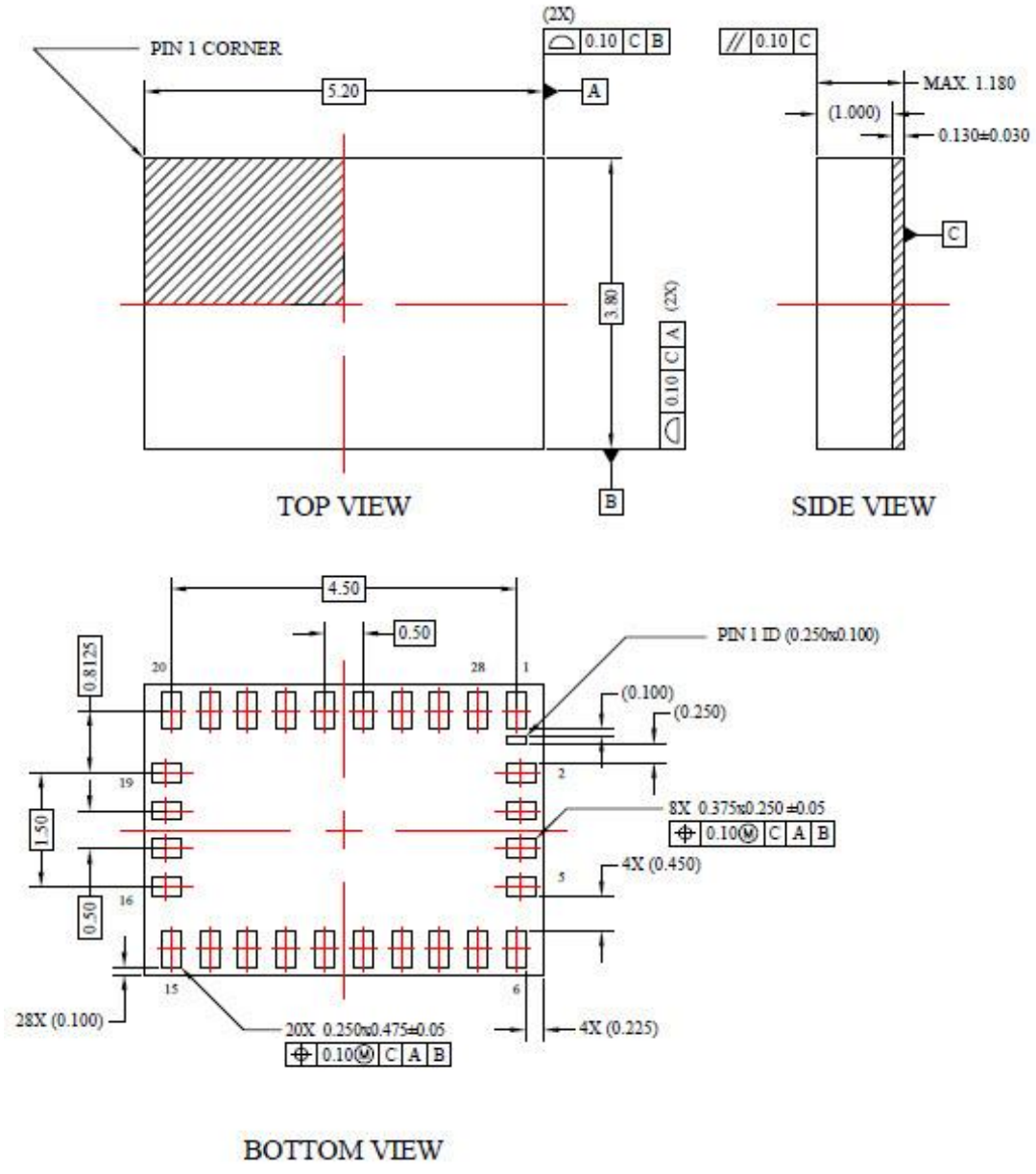
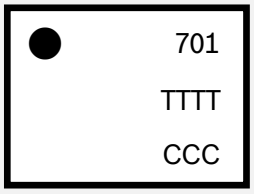


Figure 13: Outline dimensions

6.2 Marking

Table 6-1: Marking of mass production parts

Labeling	Name	Symbol	Remark
	Pin 1 identifier	●	---
	Product number	701	3 numeric digits, internal identification for product type
	Second Row	T	Internal use
	Third Row	C	Numerical counter

6.3 Soldering Guidelines

The moisture sensitivity level of the BNO055 sensors corresponds to JEDEC Level 1, see also

- IPC/JEDEC J-STD-020C "Joint Industry Standard: Moisture/Reflow Sensitivity Classification for non-hermetic Solid State Surface Mount Devices"
- IPC/JEDEC J-STD-033A "Joint Industry Standard: Handling, Packing, Shipping and Use of Moisture/Reflow Sensitive Surface Mount Devices"

The sensor fulfils the lead-free soldering requirements of the above-mentioned IPC/JEDEC standard, i.e. reflow soldering with a peak temperature up to 260° C.

6.4 Handling instructions

Micromechanical sensors are designed to sense acceleration with high accuracy even at low amplitudes and contain highly sensitive structures inside the sensor element. The MEMS sensor can tolerate mechanical shocks up to several thousand g's. However, these limits might be exceeded in conditions with extreme shock loads such as e.g. hammer blow on or next to the sensor, dropping of the sensor onto hard surfaces etc.

We recommend avoiding g-forces beyond the specified limits during transport, handling and mounting of the sensors in a defined and qualified installation process.

This device has built-in protections against high electrostatic discharges or electric fields (e.g. 2kV HBM); however, anti-static precautions should be taken as for any other CMOS component. Unless otherwise specified, proper operation can only occur when all terminal voltages are kept within the supply voltage range. Unused inputs must always be tied to a defined logic voltage level.

For more details on recommended handling, soldering and mounting please contact your local Bosch Sensortec sales representative and ask for the “Handling, soldering and mounting instructions” document.

6.5 Tape and reel specification

The BNO055 is shipped in a standard cardboard box. For details please refer to the 'Shipment packaging details' document.

6.6 Environmental safety

The BNO055 sensor meets the requirements of the EC restriction of hazardous substances (RoHS and RoHS2) directive, see also:

Directive 2002/95/EC of the European Parliament and of the Council of 27 January 2003
on the restriction of the use of certain hazardous substances in electrical and electronic equipment.

6.6.1 **Halogen content**

The BNO055 is halogen-free. For more details on the analysis results please contact your Bosch Sensortec representative.

6.6.2 **Internal package structure**

Within the scope of Bosch Sensortec's ambition to improve its products and secure the mass product supply, Bosch Sensortec qualifies additional sources (e.g. 2nd source) for the LGA package of the BNO055.

While Bosch Sensortec took care that all of the technical packages parameters are described above are 100% identical for all sources, there can be differences in the chemical content and the internal structural between the different package sources.

However, as secured by the extensive product qualification process of Bosch Sensortec, this has no impact to the usage or to the quality of the BMNO55 product.

7. Legal disclaimer

7.1 Engineering samples

Engineering Samples are marked with an asterisk (*) or (e) or (E). Samples may vary from the valid technical specifications of the product series contained in this data sheet. They are therefore not intended or fit for resale to third parties or for use in end products. Their sole purpose is internal client testing. The testing of an engineering sample may in no way replace the testing of a product series. Bosch Sensortec assumes no liability for the use of engineering samples. The Purchaser shall indemnify Bosch Sensortec from all claims arising from the use of engineering samples.

7.2 Product use

Bosch Sensortec products are developed for the consumer goods industry. They may only be used within the parameters of this product data sheet. They are not fit for use in life-sustaining or security sensitive systems. Security sensitive systems are those for which a malfunction is expected to lead to bodily harm or significant property damage. In addition, they are not fit for use in products which interact with motor vehicle systems.

The resale and/or use of products are at the purchaser's own risk and his own responsibility. The examination of fitness for the intended use is the sole responsibility of the Purchaser.

The purchaser shall indemnify Bosch Sensortec from all third party claims arising from any product use not covered by the parameters of this product data sheet or not approved by Bosch Sensortec and reimburse Bosch Sensortec for all costs in connection with such claims.

The purchaser must monitor the market for the purchased products, particularly with regard to product safety, and inform Bosch Sensortec without delay of all security relevant incidents.

7.3 Application examples and hints

With respect to any examples or hints given herein, any typical values stated herein and/or any information regarding the application of the device, Bosch Sensortec hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights or copyrights of any third party. The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics. They are provided for illustrative purposes only and no evaluation regarding infringement of intellectual property rights or copyrights or regarding functionality, performance or error has been made.

8. Document history and modifications

Rev. No	Chapter	Description of modification/changes	Date
0.1		Initial version	2013-09-02
0.2		Completely revised version (BMF055 added)	2013-10-15
0.9		Preliminary version with feature set of Firmware version 0.2.B.0	2014-04-25
1.0		Complete review	2014-07-11
1.1	3	Rearrangement of subsections in chapter 3 for better readability.	2014-11-05
	3.3	Table 3.1 is updated for better readability and all the operation modes are elaborated	
	3.11	Chapter on calibration included	
	3.7, 3.10	Update	
	4.2	The default value of the UNIT_SEL register is updated	
	4.6	I2C communication example figures are updated.	
	5.1, 5.2, 5.3, 5.4	Included table 5.1 Pin description. Connection diagram updated	
1.2	5	Updated pin description and connection diagram	2014-11-30
	6.1	Updated outline dimensions	
	6.2	Chapter removed and the respective information is updated in the Handling, soldering and mounting instructions application note.	

Bosch Sensortec GmbH
Gerhard-Kindler-Strasse 8
72770 Reutlingen / Germany

contact@bosch-sensortec.com
www.bosch-sensortec.com

Modifications reserved | Printed in Germany
Specifications subject to change without notice
Document number: BST-BNO055-DS000-11
Revision 1.1 201408

MDD10A
Dual Channel 10A DC Motor Driver

User's Manual

V2.0

June 2017

INDEX

1. INTRODUCTION/OVERVIEW

MDD10A is the dual channel version of MD10C which is designed to drive 2 brushed DC motors with high current up to 10A continuously. Just like MD10C, the MDD10A also supports locked-antiphase and sign-magnitude PWM signal. It is also using full solid state components which result in faster response time and eliminate the wear and tear of the mechanical relay.

*** MDD10A being shipping after Jan 2017 are Rev2.0 which have upgraded to support maximum motor voltage up to 30V.**

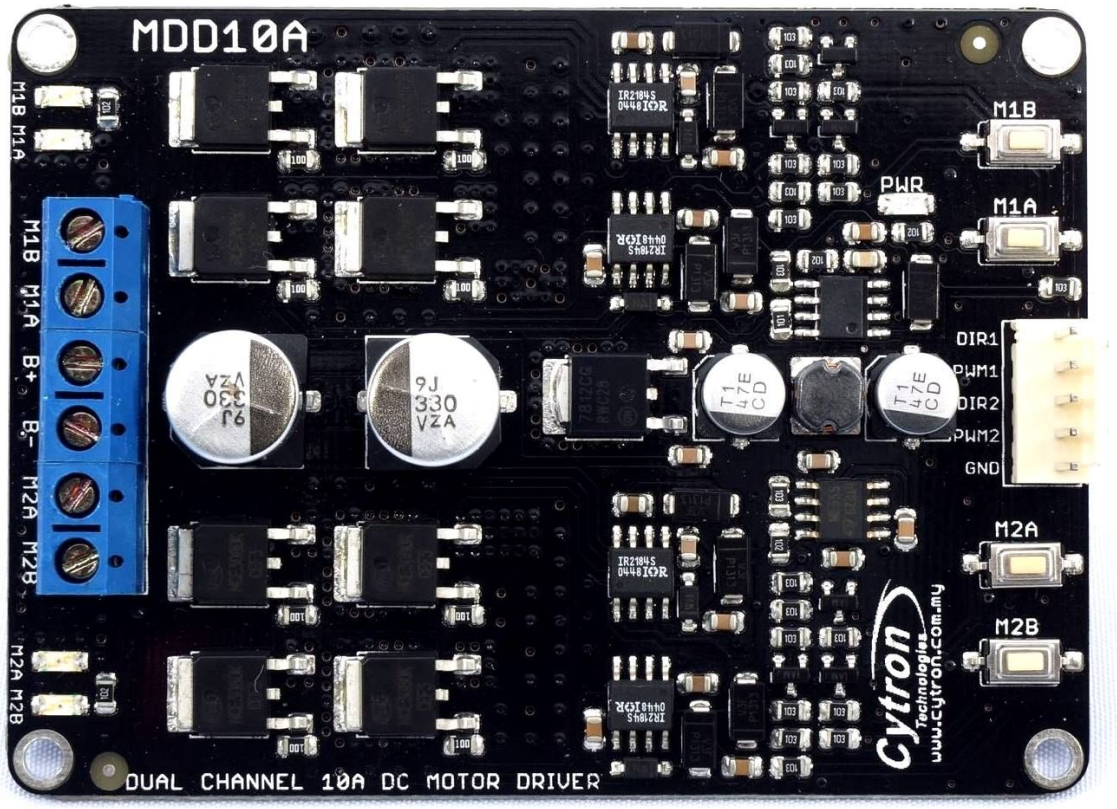
MDD10A has been designed with the capabilities and features of:

- Bi-directional control for 2 brushed DC motors.
- Support motor voltage ranges from 5V to ~~25V~~ 30V (Rev2.0).
- Maximum current up to 10A continuous and 30A peak (10 second) for each channel.
- Solid state components provide faster response time and eliminate the wear and tear of mechanical relay.
- Fully NMOS H-Bridge for better efficiency and no heat sink is required.
- Speed control PWM frequency up to 20KHz (Actual output frequency is same as input frequency).
- Support both locked-antiphase and sign-magnitude PWM operation.
- Support TTL PWM from microcontroller, **not PWM from RC receiver**.
- Onboard push button to control the motor manually.
- Dimension: **84.5mm x 62mm**

Note: Please use battery when driving inductive load such as DC brush motor. Due to the protection circuit of most Power supply (switching), it will shut down when regenerative current from motor is detected.

2. PACKING LIST

Please check the parts and components according to the packing list. If there are any parts missing, please contact us at sales@cytron.com.my immediately.



1. 1 x MDD10A Dual Channel 10A DC Motor Driver
2. 1 x 2510 PCB Connector - 5 Ways (Female)
3. 5 x 2510 Terminal Pin
4. User's manual can be downloaded from the product page under Useful Links

3. PRODUCT SPECIFICATION AND LIMITATIONS

Absolute Maximum Rating

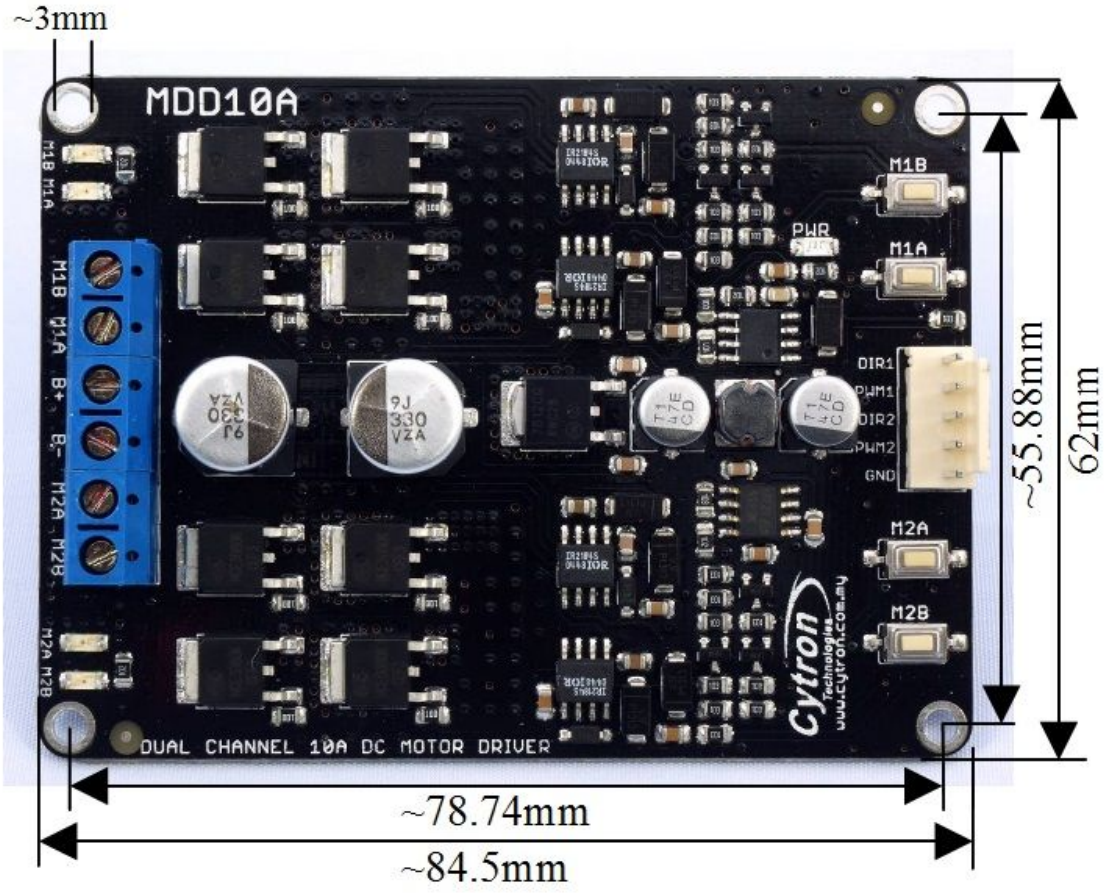
No.	Parameters	Min	Typical	Max	Unit
1	Power Input Voltage***	5	-	30	V
2	I_{MAX} (Maximum Continuous Motor Current)*	-	-	10	A
3	I_{PEAK} – (Peak Motor Current) **	-	-	30	A
4	V_{IOH} (Logic Input – High Level)	3	-	5.5	V
5	V_{IOL} (Logic Input – Low Level)	0	0	0.5	V
6	Maximum PWM Frequency	-	-	20	KHz

*** Tested in room temperature at 25°C**

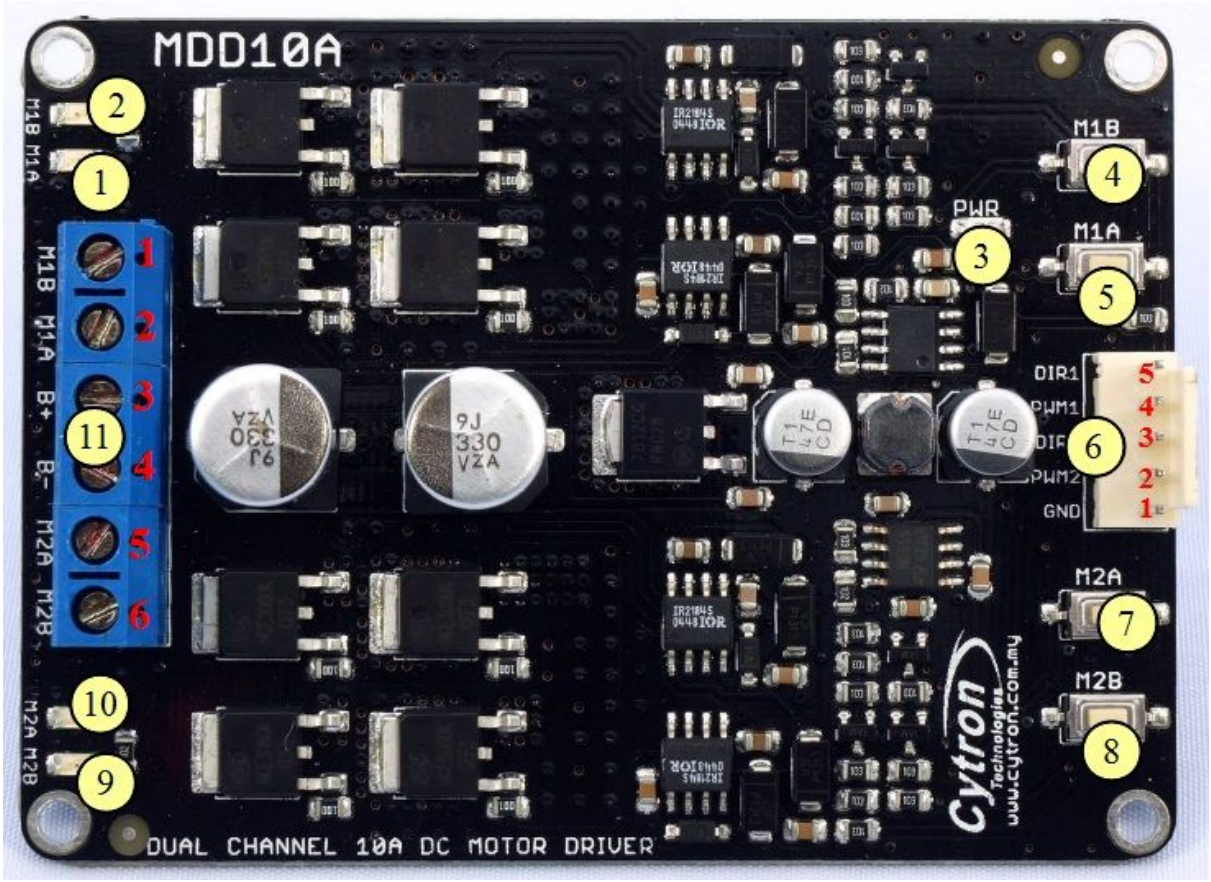
***** Must not exceed 10 seconds.***

****** Rev2.0 has upgraded to support 30V maximum power input voltage***

4. DIMENSION



5. BOARD LAYOUT



1. Red LED M1A – Turns on when the output M1A is high and output M1B is low. Indicates the current flows from output M1A to M1B.

2. Red LED M1B – Turns on when the output M1A is low and output M1B is high. Indicates the current flows from output M1B to M1A.

3. Green LED – Power LED. Should be on when the board is powered on.

4. Test Button M1B – When this button is pressed, current flows from output M1B to M1A and motor will turn CCW (or CW depending on the connection).

5. Test Button M1A – When this button is pressed, current flows from output M1A to M1B and motor will turn CW (or CCW depending on the connection).

6. Input

Pin No.	Pin Name	Description
1	GND	Ground
2	*PWM2	PWM input for speed control (Motor 2)
3	DIR2	Direction input (Motor 2)
4	*PWM1	PWM input for speed control (Motor 1)
5	DIR1	Direction input (Motor 1)

*Note that it is not for RC PWM

The truth table for the control logic for motor 1 and motor 2 are as follow:

PWM	DIR	Output A	Output B
Low	X(Don't care)	Low	Low
High	Low	High	Low
High	High	Low	High

7. Test Button M2A – When this button is pressed, current flows from output M2A to M2B and motor will turn CW (or CCW depending on the connection).

8. Test Button M2B – When this button is pressed, current flows from output M2B to M2A and motor will turn CCW (or CW depending on the connection).

9. Red LED M2B – Turns on when the output M2A is low and output M2B is high. Indicates the current flows from output M2B to M2A.

10. Red LED M2A – Turns on when the output M2A is high and output M2B is low. Indicates the current flows from output M2A to M2B.

11. Terminal Block – Connect to motor and power source.

Pin No	Pin Name	Description
1	Motor 1 Output B	Connect to motor 1 terminal B
2	Motor 1 Output A	Connect to motor 1 terminal A
3	POWER +	Positive Supply (positive terminal of battery)
4	POWER -	Negative Supply (negative terminal of battery)
5	Motor 2 Output A	Connect to motor 2 terminal A
6	Motor 2 Output B	Connect to motor 2 terminal B

Note: Please use battery when driving inductive load such as DC brush motor. Due to the protection circuit of most Power supply (switching), it will shut down when regenerative current from motor is detected. Even if the case where switching power supply is needed, a battery with the same rated voltage need to be parallel with the power supply to absorb the regenerative current.

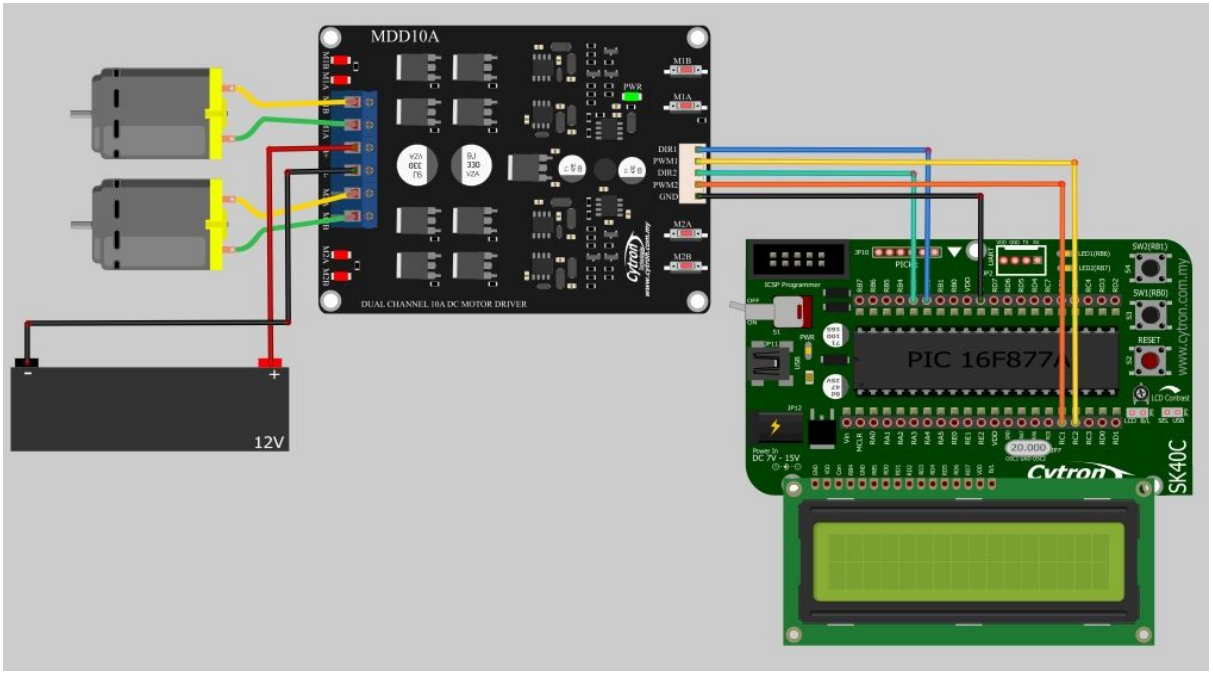
6. GETTING STARTED

6.1 Getting Started MDD10A with SK40C

MDD10A is compatible with 2 types of PWM operation, which are:

1. Sign-Magnitude PWM – For sign-magnitude PWM operation, 2 control signals are used to control the speed and direction of the motor. PWM is feed to the PWM pin to control the speed while DIR pin is used to control the direction of the motor.

2. Locked-Antiphase PWM – For locked-antiphase PWM operation, only 1 control signal is needed to control the speed and direction of the motor. PWM pin is connected to logic high while the DIR pin is being feed with the PWM signal. When the PWM signal has 50% duty cycle, the motor stops running. If the PWM has less than 50% duty cycle, the motor will turn CW (or CCW depending on the connection). If the PWM signal has more than 50% duty cycle, motor will turn CCW (or CW depending on the connection).



6.2 Getting Started MDD10A with Arduino

Check the tutorial here for the interface of MDD10A with Arduino, example sketch can be downloaded at the end of tutorial.

7. WARRANTY

- Product warranty is valid for 12 months.
- Warranty only applies to manufacturing defect.
- Damage caused by misuse is not covered under warranty.
- Warranty does not cover freight cost for both ways.

Prepared by:
Cytron Technologies Sdn. Bhd.
No. 1, Lorong Industri Impian 1,
Taman Industri Impian,
14000 Bukit Mertajam,
Penang, Malaysia.

Tel: +604-548 0668
Fax: +604-548 0669

URL: www.cytron.com.my
Email: support@cytron.com.my
sales@cytron.com.my

Appendix C: Software Tests and Charts

Test	Details	Outcome
Test software algorithm for IMU when correcting Robot's path	This test was to see how well the IMU did when serving as the primary navigator of robot. We made robot go through full navigation of 6 parking stalls and compared to pre-laid chalk lines.	The error of the lines was within 6 inches of the chalk lines we used as reference.
Test software algorithm for IMU when Robot's turning	Our robot was required to make accurate, 90 degree turns. In this test, we programmed the robot to turn 90 degrees and visually verified whether it was doing so.	The results were we found the robot always turned slightly off. The error was within a degree.
OpenCV	Installing openCV on Rpi	Successful

Figure 1: Software Tests and Res

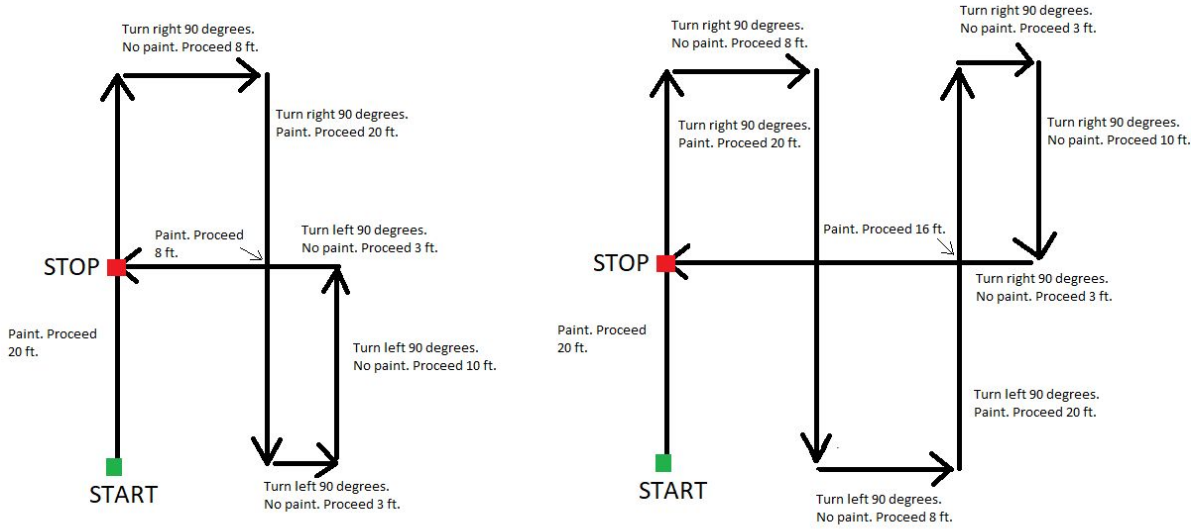


Figure 2: Software Painting Algorithm

Appendix E: Vendor Contacts

Intel:

Findley, Geof - geof.findley@intel.com
Benavides, Tony - tony.benavides@intel.com
Goles, John R - john.r.goles@intel.com
Hough, Shannon C - shannon.c.hough@intel.com
Phillips, Samuel U - samuel.u.phillips@intel.com
Cantzler, Richard M - richard.m.cantzler@intel.com
Sethi, Shruti A - shruti.a.sethi@intel.com

Parallax:

Parallax Inc
599 Menlo Drive, Ste.100,
Rocklin, CA 95765 USA
Phone: 888-512-1024

Absalom Yemane
957 Moonlit Way, Folsom, CA 95630 absalomyemane@gmail.com • 916.671.9670

OBJECTIVE:
The goal is to obtain an engineering internship and gain practical experience in the field.

EDUCATION:
In progress: **B.S. Electrical and Electronic Engineering (Emphasis in Control Systems and Analog/Digital Circuitry) • CSUS GPA: 3.41**
California State University, Sacramento • Graduating May 2018

RELATED COURSES:
- Introductory to Circuit Analysis
- Network Analysis
- Electronics I
- Electronics II
- Physical Electronics
- Electromechanical Conversion
- Introduction to Microprocessors
- Introduction to Feedback Systems
- Electronics II Laboratory
- Intro. To Modern Communication Systems

KNOWLEDGE AND SKILLS:
Communication/Organization:
- Exceptional team skills
- Exceptional organization skills
- Strong Technical Writing skills
- Adapt to new situations quickly
- Strong Leadership skills

Software:
• ADS • C++ • MATLAB • MS Office • MultiSim • C • PSPICE • Arduino • Raspberry Pi

WORK EXPERIENCE:
Student Tutor, Folsom Lake College Tutoring Center 1/15 – Present
Teaches students' fundamental concepts in mathematics, computer programming, and physics.
Student Engineer, City of Sacramento 5/16 – 8/16
Prepare as built drawings for various electrical engineering projects.
Assists with meetings, including preparation of meeting agendas and materials.
Prepare specifications for various electrical engineering projects using Word and AutoCAD.
Academic Mentor, Your First Choice Tutoring 1/17 – Present
Teaches students' fundamental concepts in mathematics, computer programming, and physics in a one-on-one setting.

PROJECTS:
Self-Regulating Water Tank System:
Based on two desired variables, myself and three others created a three-bucket water system that maintained a desired water temperature and water height continuously. All that was described was accomplished with a Propeller microcontroller as the central component to our system.

AWARDS/CLUB AFFILIATION
ECS Dean's Honor Roll Student 08/15 – Present
Maintained an exceptional grade point average in the Engineering and Computer Science department at Sacramento State University.
Industry Liaison Officer, IEEE CSUS Student Chapter 08/16 – 12/16
Acts as liaison and IEEE representative with companies from industry. Lead efforts to provide tours, networking, and professional events for IEEE members.
President Elect, IEEE CSUS Student Chapter 1/17 – Present
Serves as an assistant to the president in preparation to take over as president for the following term. Follows up with all other officers to maintain organization for all IEEE events during the semester.

DAVID RYAN JR

ELECTRONICAL & ELECTRONICS ENGINEER | DIGITAL/ANALOG CONTROLS

SUMMARY

Electrical engineering student in digital/analog systems and controls engineering.

SKILLS

Manufacturing
Assembly
Inventory Management
Mechatronics
Robotics
Soldering
Semiconductors
Testing/Troubleshooting
Networking
Public Speaking
Customer Service

EXPERIENCE

ELECTRICAL ENGINEER • ROCKY RIDGE WIRELESS • JULY 2016 – PRESENT
Renewable power system design. Network maintenance and communication technology. Tower climbing and installation. Customer care and tech support. Introduction to ladder logic control systems for public utility facilities.

ASSEMBLY TECHNICIAN • KENT JOHNSON CONSULTING SERVICES • JAN 2011 – FEB 2014
Experience in the hardware assembly and testing of various electrical systems, specifically in the area of microwave electronics. Circuit board stuffing, building wiring harnesses, and inventory control.

EDUCATION

ASSOCIATES OF ARTS- MATH AND SCIENCE • DEC 2014 • FOLSOM LAKE COLLEGE
Certified UC and CSU applied communications specialist.

BACHELOR OF SCIENCE- EEE • MAR 2018 • CALIFORNIA STATE UNIVERSITY SACRAMENTO
Officer of Newman club on campus. Team lead in a renewable energy storage research team.

SMITTYRYAN1.2@GMAIL.COM

530-417-4203

LINKEDIN.COM/IN/DAVID-RYAN-JR-293A3A56

Eric C. Klenner

ericklennerperiod3@gmail.com
(916) 706-6548

OBJECTIVE ***To obtain a position where I can utilize and continually develop my skills in information assurance and network security***

EDUCATION

Bachelor of Science in Computer Engineering — **GPA:** 3.5
Minor in Mathematics — **Expected:** May 2018
Certification in Cyberdefense and Operations
Certification in Systems Programming
California State University, Sacramento

Related Courses:

Computer Networks and the Internet	*Operating System Principles and Practice*	*Compiler Construction*
Cryptography	*Advanced Computer Organization*	*Operating System Pragmatics*
Network Security	*Computer Forensics*	

KNOWLEDGE AND SKILLS

Programming Languages: Java, C, Python, Verilog, VHDL, x86 Assembly, MIPS
Operating Systems: Windows 95, 98, 2000, XP, Vista, 7, 8, 8.1, 10, Mac OS, Linux
Software: DOS, Putty, MobaXterm, Ultra VNC, VMware, VirtualBox, Visual Studio, Eclipse, JGrasp, Quartus, Xilinx Vivado Suite, Cadence Virtuoso, Multisim, PSPICE, Microsoft Office Suite, Wireshark, NMAP, Splunk, JIRA, Ollydbg, IDAPro

Personal Skills

- Flexible, can work efficiently solo or on a team
- Articulate communicator, written and verbal
- Excellent priority scheduling
- Analytical and critical thinking
- Versatile, able to acquire new skills rapidly
- Able to approach technical problems from both a hardware and software perspective

PROJECTS

Compiler Construction: Constructed a compiler for the Baby C language — *Fall 2017*
Operating System: Constructed a basic C-based operating system with a partner — *Spring 2017*
Forensic Examination: Collected, analyzed, and reported on an image of a laptop computer — *Spring 2016*
AES Program: Wrote a program in C to encrypt and decrypt files using AES encryption in counter mode — *Fall 2016*
16-bit CPU: With a partner, constructed a CPU in Verilog for a unique 16-bit instruction set — *Spring 2016*

ACTIVITIES AND AWARDS

- NSF Scholarship for Service Award — *Sep 2016*
- Member of Tau Beta Pi Engineering Honor Society — *Dec 2015*
- Member of Phi Kappa Phi Honor Society — *April 2018*
- Computer Engineering Department Outstanding Student Award — *May 2018*
- Dean's Honor List — Spring 2015, *Fall 2015, Fall 2016, Spring 2017*
- CCDC Team Captain at CSU Sacramento — *Fall 2017-Spring 2018*

CERTIFICATIONS

- Splunk Certified User — *Jul 2017*
- CompTIA Security+ — *Sep 2017*

TECHNICAL WORK EXPERIENCE

Student Intern with Network Security Operations Team *Jun 2017 – Aug 2017*

Los Alamos National Laboratory

Responsibilities

- Performed preliminary investigations into suspicious traffic on the network
- Used Splunk software to search and analyze network events including web proxy traffic, DNS, email, and WLS logs
- Recorded progress on investigations in a ticketing service.
- Researched the viability of using Salt-Proxy and Salt-SSH as an automation tool in a secure network environment

Ferishta Ansari

4 Niebaum Ct, Elk Grove, CA 95758
(916) 501-8662, FerishtaAnsari@gmail.com

OBJECTIVE

To obtain an internship position as an Electrical Engineer using demonstrated communication and interpersonal skills

SKILLS SUMMARY

- Over 4 years experience working in customer service
- Fluent in Farsi and English,
- Work well independently as well as in a team capacity
- Expert in critical thinking

EDUCATION

Cosumnes River College, Sacramento
Electrical Engineering, expected transfer May 2015
Relevant Coursework: Computer information Science Programming C++, Physics (Electricity and Magnetism), Linear Algebra, and 3-D Calculus

Sacramento State University
B.S. degree in Electrical Engineering, expected graduation date, May 2018
Relevant Coursework: Advanced Logic Design (VHDL and Verilog), Electronics 1, Signals and Systems

PROJECTS/PERSONAL EXPERIENCE

- Created a code in C to output up to 100 rows of Pascal's triangle using multidimensional arrays
- Responsible for the three D printing of parts, and coding of the servo motors used for the Skittle Color Sorter designed by a team of four

PROFESSIONAL EXPERIENCE

Cosumnes River College, Sacramento
SI math Leader

- Sat in during lectures to learn the professor's teaching techniques
- Used these teaching techniques during sessions to maintain consistency for the students
- Provided a collaborative teaching environment to ensure students participate
- Improved student's test scores by at least one letter grade

Best Buy, Elk Grove, CA 2013-2014
Tablet's sales consultant

- Promptly Greeted customers, and used recommendation sheets to build rapport
- Asked questions based on the customer's life style to recommend the perfect tablet
- Offered the right accessories, protection plans, and credit card to match the customer's lifestyle

Cosumnes River College, **Financial Aid Office,** Sacramento, CA 2012 - 2013
Student Assistant

- Assisted students via telephone with general questions regarding the financial aid process
- Accepted proper documentation from students to complete the process of their financial aid file
- Acquired skills in computer related programs such as PeopleSoft and PowerFaids
- Practiced secretarial duties such as filing and scanning
- Translated from English to Farsi for Afghani and Iranian students

Cosumnes River College, Sacramento, CA 2012 - 2015

Professor's Assistant

- Assisted the college professor in grading elementary and intermediate algebra assignments
- Recorded the data on a recording sheet created by the professor
- Ensured accurate transmission
- Made notes on students' assignments by explaining the proper techniques to solve problems

HomeTown Buffet, Elk Grove, CA 2012

Cashier

- Enthusiastically greeted customers
- Properly rung more than 300 customers at the register daily
- Maintained a high percentage of beverage sales

Hometown Buffet

Waitress

- Enthusiastically greeted customers 2010
- Picked up unwanted plates from customers
- Ensured tables were wiped off as soon as the customers leave
- Constantly swept the floors to maintain a clean restaurant environment

Chanchiew Saeteurn

922 Briarwood Drive, West Sacramento, CA 95691 • Cell: 9167150313 • s.chanchiew@gmail.com

Qualifications

Electrical and electronics engineering student with the motivation to go above and beyond what is neccessary to get the job done. Having the skills to design and simulate electronic circuits with software such as matlab, simulink, pspice, multisim, and advance design system. Proficient in C programming language, familiar with Java, and having the skills neccessary to learn a language quickly.

Skills

- Programming Microcontrollers in C such as arduino, Raspberry Pi, Parallax, and Microchip.
- Familiar with Java and Windows Operating systems
- Coding C in Matlab and running simulations on simulink
- Running simulations on ADS (Advance Design System) - MicroStrip Design
- P-Spice/Multisim, designing circuits and anaylzing frequency ranges
- Proficient in Microsoft Word, Powerpoint, and Excel
- Technical support
- Power systems
- Time management
- 3D designing
- Rapid Prototyping

Work History

Supervisor , 10/2012 to Current
Pizza Guys Inc. – 800 Harbor Blvd, West Sacramento, CA, 95691
Time managing schedules and shifts.
Closing out registers at the end of the day.
Enter data to excel at the end of the day.
Conduct interviews.

Education

Undergraduate: Electrical and Electronics Engineering, Current
California State University of Sacramento -

- IEEE member
- Micromouse Competition Fall 2017 - First place
- Continuing coursework in electronics and communications
- Critical Electrical Systems coursework
- Continuing coursework in C, C++, JavaScript and MATLAB
- Overall GPA 3.034