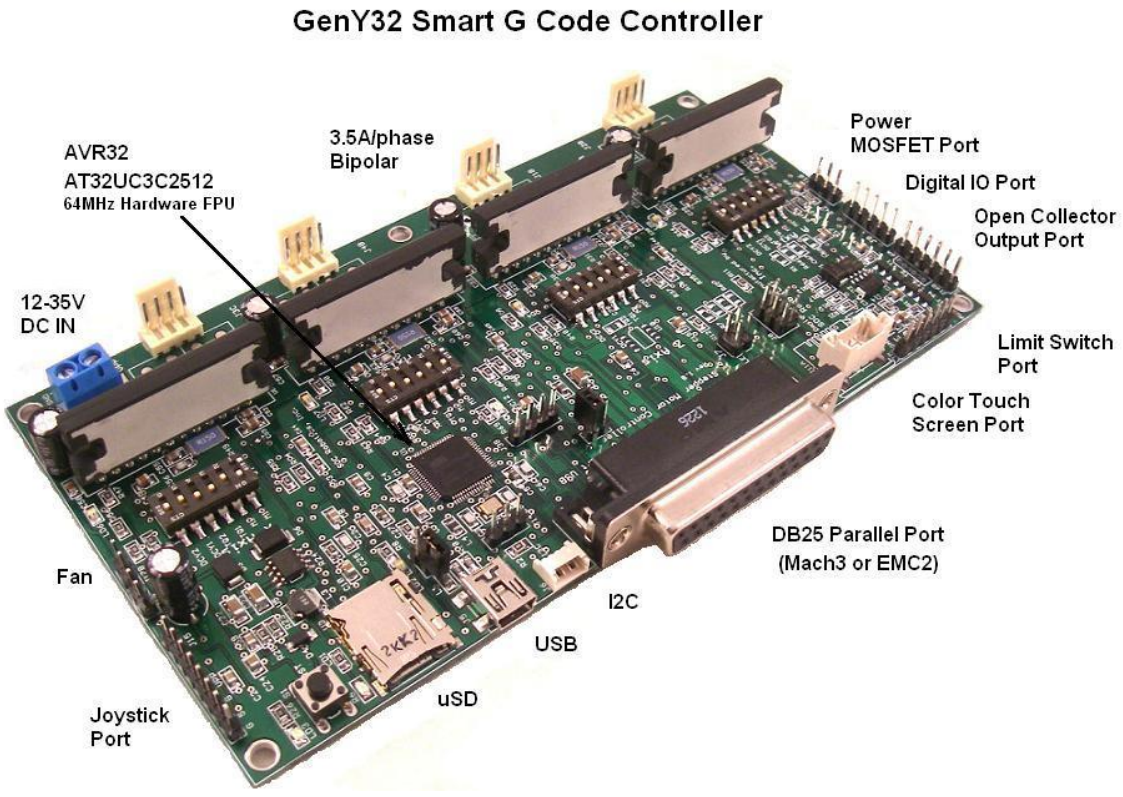
GenY32 Smart G Code 4-Axis Controller

4-Axis, 3.5A Bipolar Drivers
AT32UC3C2512 32bit AVR32 Processor

Technical Reference Manual

PCB Rev 1.0



www.soc-robotics.com

Warranty Statement

SOC Robotics warrants that the Product delivered hereunder shall conform to the applicable SOC Robotics Data Sheet or mutually agreed upon specifications and shall be free from defects in material and workmanship under normal use and service for a period of 30 days from the applicable date of invoice. Products that are "samples", "design verification units", and/or "prototypes" are sold "AS IS," "WITH ALL FAULTS," and without a warranty. If, during such warranty period, (i) SOC Robotics is notified promptly in writing upon discovery of any defect in the goods, including a detailed description of such defect; (ii) such goods are returned to SOC Robotics, DDP SOC Robotics facility accompanied by SOC Robotics Returned Material Authorization form; and (iii) SOC Robotics examination of such goods discloses to SOC Robotics satisfaction that such goods are defective and such defects are not caused by accident, abuse, misuse, neglect, alteration, improper installation, repair, improper testing, or use contrary to any instructions issued by SOC Robotics. SOC Robotics shall (at its sole option) either repair, replace, or credit Buyer the purchase price of such goods. No goods may be returned to SOC Robotics without SOC Robotics Returned Material Authorization form. Prior to any return of goods by Buyer pursuant to this Section, Buyer shall afford SOC Robotics the opportunity to inspect such goods at Buyer's location, and any such goods so inspected shall not be returned to SOC Robotics without its prior written consent. SOC Robotics shall return any goods repaired or replaced under this warranty to Buyer transportation prepaid, and reimburse Buyer for the transportation charges paid by Buyer for such goods. The performance of this warranty does not extend the warranty period for any goods beyond that period applicable to the goods originally delivered.

THE FOREGOING WARRANTY CONSTITUTES SOC ROBOTICS EXCLUSIVE LIABILITY, AND THE EXCLUSIVE REMEDY OF BUYER, FOR ANY BREACH OF ANY WARRANTY OR OTHER NONCONFORMITY OF THE GOODS COVERED BY THIS AGREEMENT. THIS WARRANTY IS EXCLUSIVE, AND IN LIEU OF ALL OTHER WARRANTIES. SOC ROBOTICS MAKES NO OTHER WARRANTIES, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING WITHOUT LIMITATION ANY WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. THE SOLE AND EXCLUSIVE REMEDY FOR ANY BREACH OF THIS WARRANTY SHALL BE AS EXPRESSLY PROVIDED HEREIN.

Limitation on Liability

Notwithstanding anything to the contrary contained herein, SOC Robotics shall not, under any circumstances, be liable to Buyer or any third parties for consequential, incidental, indirect, exemplary, special, or other damages. SOC Robotics total liability shall not exceed the total amount paid by Buyer or SOC Robotics hereunder. SOC Robotics shall not under any circumstances be liable for excess costs of re-procurement.

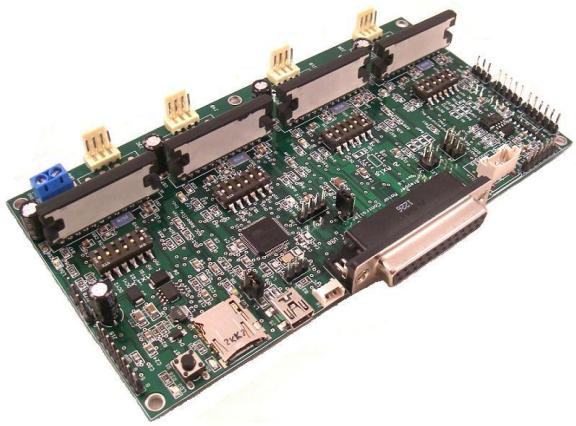
Table of Contents

Introduction

Features:

- 4-Axis Stepper motor controller – 3.5A/phase bipolar driver TB6560
- High performance onboard 32bit processor – AT32UC3C2512
- USB 2.0 interface
- Embedded G Code controller software (GStep) built-in
- Optional GRBL port available
- Standard female DB25 parallel port interface
- Accepts uSD cards up to 32G
- Five limit switch inputs
- Four auxiliary outputs with open collector and digital outputs
- 2 Ch high current 3A Power Mosfet outputs
- Green LED indicates power status
- Green/Red Status LED's
- Input logic power protection and reverse polarity protection
- Transient Voltage Suppression Diodes on Motor Signals
- Joystick analog input port
- Processor reprogramming via on chip DFU bootloader
- Smart Color Touch screen communications port
- I2C port
- 18-35VDC @ 80ma Power input for logic and motor
- Compact form factor (5.4x2.2in)

GenY32 Smart G Code Controller



Hardware

GenY32 combines the best legacy parallel port design with the latest high performance embedded G Code technology with USB interface into a single unit. The parallel port ensures existing Mach3/EMC2 parallel port applications run immediately while the onboard processor provides a migration path to next generation applications that communicate with the board via the USB interface or execute G Code files from the onboard uSD adapter eliminating the need for a desktop controller altogether. The uSD adapter allows execution of stored G Code without a desktop connection and the new SOC Color Touch Screen pendant allows user control of G Code file selection and board operation.

GenY32 is a 4-axis stepper motor controller with 4 auxiliary outputs and 5 limit switch inputs. The onboard processor monitors an analog joystick port, uSD and Smart Color Touch Screen pendant port.

GenY32 has a jumper-selected feature that allows Auxiliary outputs OUT3 and OUT4 to be used to control TB6560 driver chips Reset and Enable inputs. This allows the desktop to set the TB6560 to a known initial state and to enable or disable motor power.

GenY32 has four TB6560 1/16th Microstepping Bipolar 3.5A per phase stepper motor drivers that convert step and direction signals to the appropriate high voltage stepper motor drive signals. Motor voltage can vary from 10-34VDC. A DIP switch sets step mode (full, half, 1/8th and 1/16th), torque level (100%, 75%, 50% and 20%) and decay mode (0%, 25%, 50% 100%).

GenY32 consumes about 80ma in active state plus the load of the stepper motors that can reach 14A if all four axis are set to maximum torque and full step.

GenY32 is compatible with Mach3 and EMC2 (LinuxCNC).

Processor and G Code Application

GenY32 has a 32bit AVR32 processor running at 64MHz. The processor runs a pre-installed G Code Interpreter application called GStep. GStep communicates with the desktop via the USB 2.0 interface as a CDC serial communications application (COMX). In order to use the CDC function a driver must be installed on the Windows desktop. The driver is included with GenY32 Release Folder and is called "atmel_devices_cdc.inf". There is a jumper next to the USB port that allows the USB connection to power the board rather than the 24V input connection. This jumper should not be installed when powering the board from a 24V source. Install the driver when Windows asks for it.

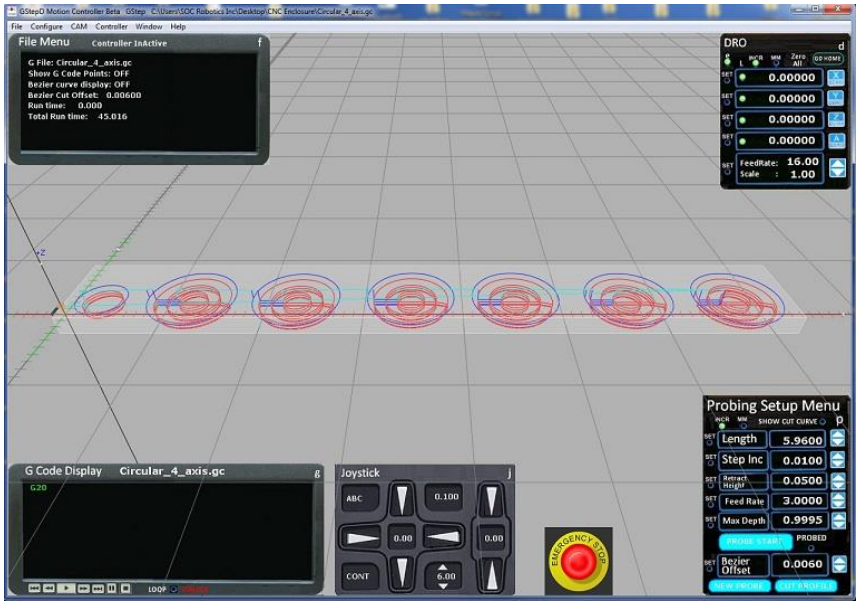
You can use a simple terminal application to communicate with GStep once the driver is installed. Baud rate, parity and data bits are not important and are ignored by the application. Note that a simple console DNC application is included in the Release Folder to drip feed G Code to the board.

When shipped from the factory GStep does not drive the step/direction signals required by the TB6560 stepper motor drivers. This is a pre-caution for those users that decide to drive the board via the parallel port. To enable GStep's control of the motor drivers enter the esc cc command at the top level menu (ie ESC key followed by cc ENTER).

GStep supports a rich set of G Code commands and has full support for linear and circular interpolation. See Chapter 6 for a full description. Although GenY32 is a four axis controller GStep is designed to be a six axis controller. GenY32 can be extended to a six axis configuration using other SOC Robotics motion control products.

Windows GUI - GStepD

GStepD, a comprehensive Windows GUI, supports real time machine simulation, G Code path display and DNC is under development and is scheduled for release in summer 2016 (beta test releases scheduled for mid May). GStepD communicates with GenY32 GStep and synchronizes its operation with the controller. Desktop joysticks are automatically integrated with the GUI and support for GenY32 uSD file management is included. GstepD is designed both as a real time interface for GenY32 and as a desktop simulator to assist with G Code debugging prior to downloading to the controller.



Windows GUI – Mach3 Plugin

A Mach3 plug-in is available to allow Mach3 to drive GenY32 via the USB port. Plugin Version 0.99 is available for download on the website.

grbl and tiny ports

grbl (v0.8c) has been ported to GenY32 and full source code can be downloaded from the web site. A port of tinyg is in the works and should be available in May 2016.

Smart Limit Switches

GenY32 is being engineered to support SOC Robotics new line of Smart Limit Switches. Smart Limit Switches communicate status information such as proximity or contact closure to the main processor allowing multiple home positions per limit switch.

Configurations

GenY32 has a few configuration options under control of the onboard processor. While in monitor mode the onboard processor can still activate control of the Power MOSFETs via the parallel port signals OUT1 and OUT2 and TB6560 Enable and Reset. See the Chapter describing GStep for more information.

1.0 GenY32 Quick Start Guide

2.1 Introduction

Before using GenY32 please read this quick start guide.

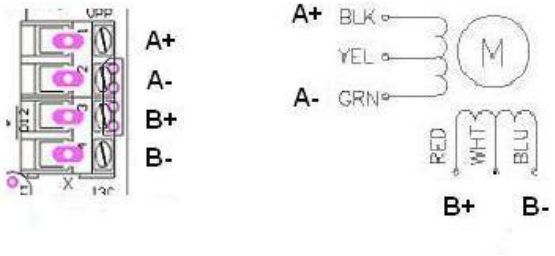
WARNING: Before connecting Motor DC Power make sure the stepper motors are wired correctly. The TB6560 drivers used on GenY32 connect to bipolar 4-wire motors. Depending on how the windings are connected the motor will turn clockwise or counterclockwise. Ensure the motor leads are securely attached – intermittent or loose connections can generate a several kilovolt back EMF spike in the motor possibly damaging the motor and/or GenY32. NEVER ATTACH or DETACH MOTOR WIRES WITH POWER APPLIED.

2.2 Wiring the Stepper Motor

The wiring of a Bipolar Stepper motor to GENY32 is shown in the diagram below. The color code used in the diagram is for a SOC Robotics **SM2006A** 2.5A Stepper Motor and may not match the color code of your particular stepper motor – check the phase (coil) configuration diagram for your stepper to determine the correct lead connection to **A, A+,B** and **B+** before attaching the stepper to GenY32. Do not connect or disconnect stepper motor leads while GenY32 is generating a step pattern – always turn the DC Motor Power Supply off first. Also never touch the open leads of the stepper motor while they are in operation.

Typical Stepper Motor Wired in BiPolar Configuration

(Color codes may vary)



Common Motor DC Connection

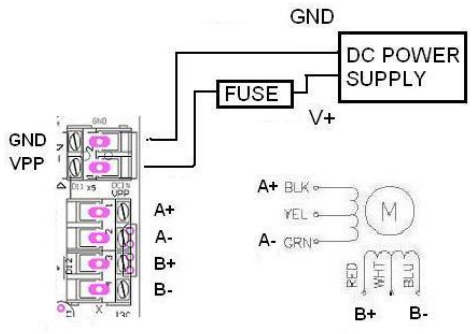
Its highly recommended to install an inline fuse between the DC Motor Power supply and GenY32 - the Power MOSFETs in the TB6560 can fail and when they do they typically fail in a shorted state. Ensure there is a good ground between the DC Motor Controller, Board Logic Power and the controlling PC – GenY32 is a non-isolated design. All attached equipment must be at a common ground potential. Do not defeat the ground lug of any AC cords.

Motor power is routed to all four TB6560 drivers on the board. The diagram below shows a typical wiring configuration. For this configuration DC Motor Power is connected to screw top terminal X1 where it is then routed to the TB6560 driver for each motor. Geny32 is provided with

four press-on connector headers with crimp connectors. When assembling the connector be sure to crimp the motor wires securely and then solder the crimped connectors. The most likely mode of failure is loose connections. Crimping is best done with needle nose pliers.

Common DC Power Supply Connection with Inline Fuse

(Color codes may vary)

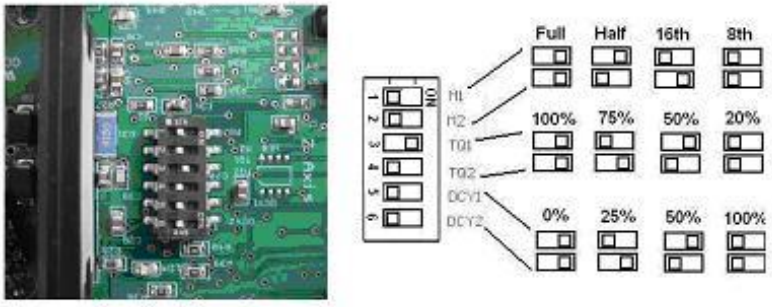


2.3 Hardware Setup

GenY32 must be configured for correct operation. If the onboard G Code controller is used then a Windows drivers must be installed. If the parallel port is used then the G Code controller must be disabled. The G Code controller is disabled when shipped from the factory and must be re-enabled by removing the DFU jumper and/or sending the ESC cc command via the USB CDC interface.

In both cases motor step mode, decay and torque are set using a DIP switch located beside each of the four TB6560 drivers. The default factory setting is 1/8th microstep, 100% decay and 75% torque. For smooth operation 1/8th microstep is recommended for most applications – this will produce smooth stepping, reduce resonance, support high-speed operation and low audible output. The torque setting should match your motors. Once again for smooth operation 75% is recommend for most NEMA23 motors in the 2.0 to 3.0A range although good operation is also achievable at 50% torque.

TB6560 DIP Switch Settings



Heat sinks are required with forced air-cooling. The metal back of each TB6560 can be commoned across all four drivers with a metal plate to assist with cooling. An optional custom CNC'd heat sink/fan assembly is available.

Parallel Port Setup

1. Set the step mode, decay and torque for each of the four drivers.
2. Install the DFU jumper to disable the onboard processor
3. Disconnect the USB power jumper.
4. Connect a DC power adapter to the Board Logic input (24-36VDC – center positive – reverse polarity protected).
5. Connect a parallel port cable between the PC and GenY32.

G Code Controller Setup

1. Attach a USB cable and power up the board.
2. Windows will ask for a driver – install the CDC driver from GenY32 Prelease folder.
3. Attach a terminal program such as Hyperterminal to the COM channel GenY32 is attached to. Baud rate is not important but set to 115,200, N,8.
4. Hit return a few times – this should return the '$' prompt. The '$' characters is the command prompt – commands are entered after the prompt and executed when the RETURN key is pressed. Make sure Control Mode is active. The sign-on message displays the state of Control mode.
5. Enter "?s" to get the signon message. If control mode is not active (default from factory) switch to control mode by entering the Escape key followed by "cc" – this will switch GenY32 into control mode.

```
GenY32 Stepper Controller - Gstep V2.05
Copyright 2013, SOC Robotics, Inc.
Type H for help menu - Control mode active

$
```

The G Code processor preference settings must be configured. The G Code processor assumes that the resolution of each axis is 0.00025in per step, there is no backlash, maximum rapid move is 20ipm and step direction polarity is P N N N. See the G Code Processor command section to understand what these setting are. The preference setting menu is entered by entering the "se" command:

```
-se
Entering Preference Mode
Type 'le' to leave - dp to display preferences - ss save settings
=
```

For the current preference settings enter "dp".

```
=dp
Preference Settings:
Baud rate = 115200
Max Feed rate (mf)= 20
Resolution [X,Y,Z,A] (rr)= 0.000125 0.000125 0.000125 0.000125
Backlash [X,Y,Z,A] (bl)= 0 0 0 0
Direction polarity [X,Y,Z,A] (pl)= P N N N
=
```

Now change any preference settings by entering a two character preference selection command followed by the new values. For example to change the resolution from 0.00025 to 0.000125 enter the following command:

```
=rr0.000125 0.000125 0.000125 0.000125
=
```

Changed preferences are not saved in FLASH until the "ss" command is entered.

GenY32 should now be ready to use. To move the X axis 1 inch enter the following G Code command at the prompt:

```
-G1 X1.00 F10.0 <CR>
-
```

Preference settings can also be stored in a configuration file on the uSD by creating a configuration file called confg.txt.

2.4 Host Software Setup

Desktop application software sends G Code commands and/or GENY32 configuration commands to GenY32 via the serial line. If commands are sent to the controller to fast the serial receive buffer will overflow so commands must be allowed to complete before sending the next command.

GenY32 expects a low to high transition on the Step input to generate a step. The Direction input causes a clockwise step or counterclockwise step depending on how the motor is wired to the drive. The direction input is sampled after the step is activated so it must remain unchanged for at least 5useconds after the step low to high transition. Step and direction inputs can be changed at the same time. Host software must the set the Direction line high or low and cause a low to high transition on a step line to cause a step. The Direction and Step inputs are pulled high with a 10K resistor. Noise reduction logic in the motor processor eliminates false steps due to noise.

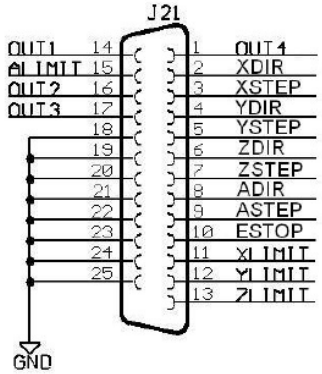


Figure 2-1. **Parallel port signal function assignment.**

Limit switch inputs are pulled high with 10K resistors – an external contact closure will pull the limit switch input low indicating a limit has been reached. Each motor processor monitors the limit switch input pertaining to its stepper motor and automatically stops the motor without requiring input from the host. Host software should interpret a low level on a limit pin as an indication that a limit has been reached.

Four Auxiliary outputs are available for host software to control. Each Auxiliary output is routed to two different connectors: Relay adapter connector J17 and open NPN transistor output

connector J18. A high level on an OUTX pin sends a high level to the relay port and causes the appropriate transistor to conduct pulling the output lead to ground on J18. Host software must be configured accordingly to control external devices.

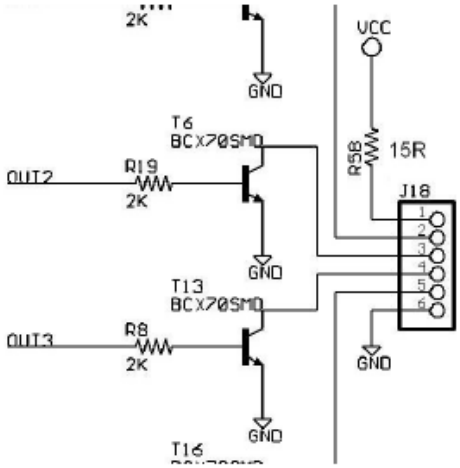


Figure 2-2. **Typical open NPN Auxiliary Output circuit on J18.**

GenY32 Detailed Description

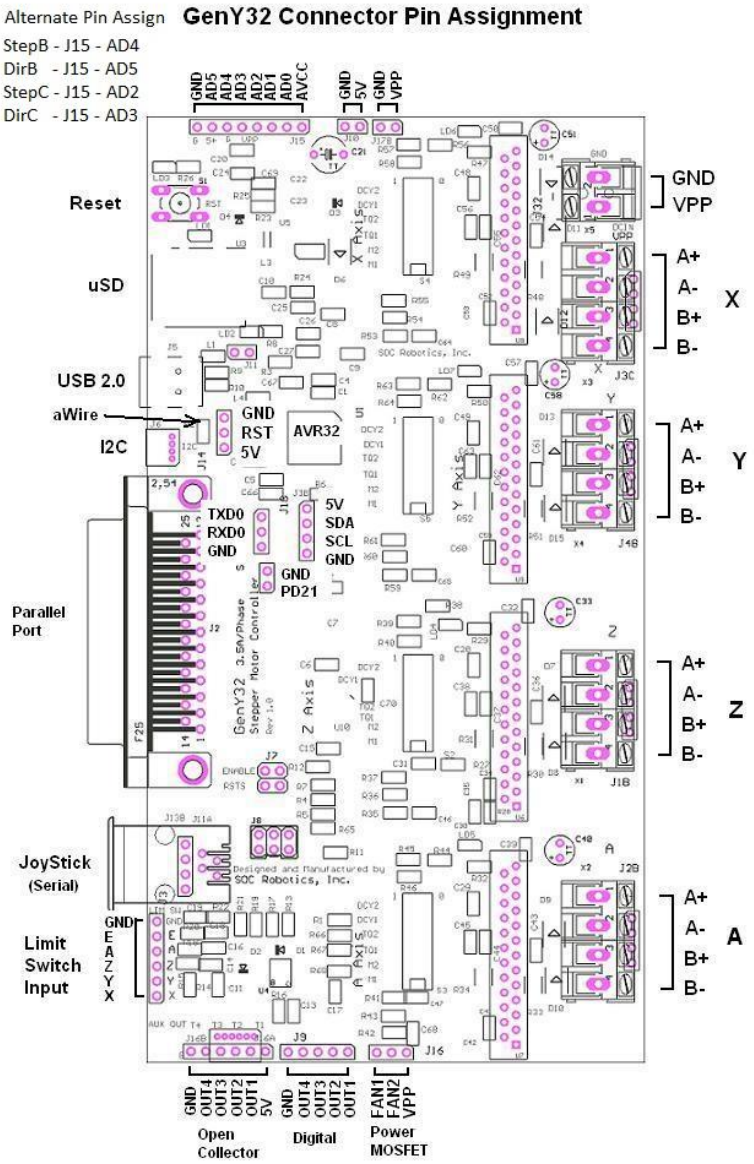
3.1 Introduction

GenY32 is a 4-Axis stepper motor controller with both a parallel port interface and onboard high performance 32bit processor running an embedded G Code interpreter. GenY32 can be controlled either by the onboard processor or the parallel port but not both at the same time. By default the onboard processor is disabled and must be enabled by the user. Once enabled the parallel port can no longer be used. Disabling the processor allow reuse of the parallel port.

GenY32 connects step and direction inputs from the PC parallel port to each of the four stepper drivers (TB6560). The correct step and direction signals for each axis must be assigned to the correct parallel port pin. For example, DIRX is assigned to pin 2, STEPX is assigned to pin 3, etc. The step signal is an active going high pulse. The circuit schematic and picture below show the mapping between Step/Direction/Limit/Out signals on the DB25 and the four axis ports, limit switch inputs and auxiliary outputs.

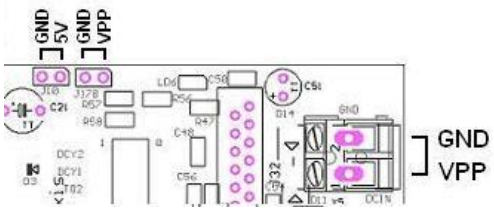
DB25 Pin Assignment

Pin	Function
1	OUT4
2	DIRX
3	STEPX
4	DIRY
5	STEPY
6	DIRZ
7	STEPZ
8	DIRA
9	STEPA
10	EStop
11	LimitX
12	LimitY
13	LimitZ
14	OUT1
15	LimitA
16	OUT2
17	OUT3
18-25	GND



3.2 Input Power

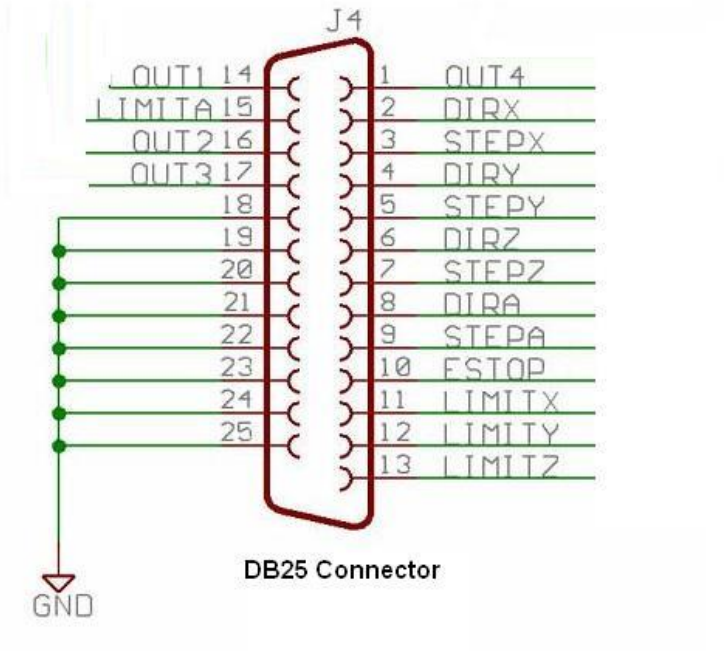
A single DC 12-35V supply powers the board and supplies power to the stepper motor drivers. The onboard switching power supply converts the 12-35VDC input to 5V used by the processor and related logic. A two pin header J17B located beside the DC input connector can supply power to a separate optional fan required to cool the TB6560 drivers. The power on J17B is the same as the DC input power. An additional two-pin header next to the fan power supplies 5V and GND for general use to power off board peripherals.



3.3 Parallel Port

GenY32 has a standard DB25 parallel port that can be used to drive the four TB6560 drivers using Mach3, EMC2 or some other parallel port desktop G Code application. GenY32 DB25 default pin assignment is shown in the diagram below.

The parallel port should only be used when the onboard processor has released control of the step/direction/limit and output lines. The board is shipped with the processor not in control of the lines for safety. If you intend to use the onboard G Code Controller then the processor must regain control of these lines. Note that once the processor gains control of these lines the parallel port cannot be used and no cable should be attached to the DB25. The processor monitors the status of the eStop input line at power up - if this line is low (ACTIVE) then the processor automatically releases control to the parallel port and enters Monitor Mode. The processor stays in Monitor Mode until instructed to enter Control Mode via the USB interface.

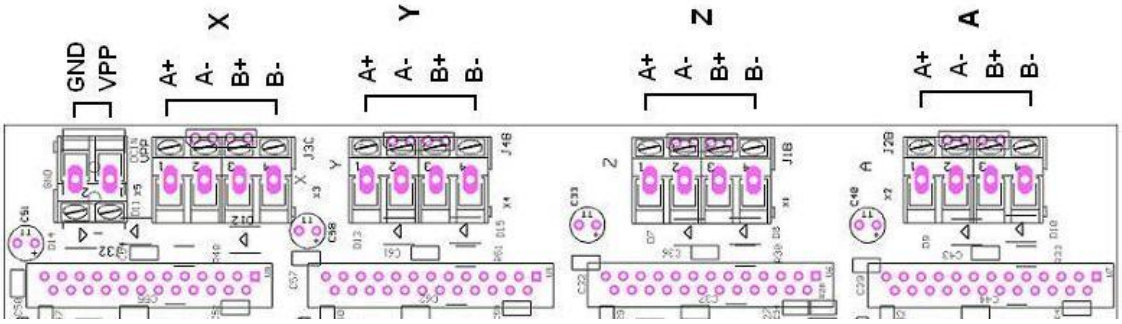


DB25 Pin Assignment

Pin	Function
1	OUT4
2	DIRX
3	STEPX
4	DIRY
5	STEPY
6	DIRZ
7	STEPZ
8	DIRA
9	STEPA
10	EStop
11	LimitX
12	LimitY
13	LimitZ
14	OUT1
15	LimitA
16	OUT2
17	OUT3
18-25	GND

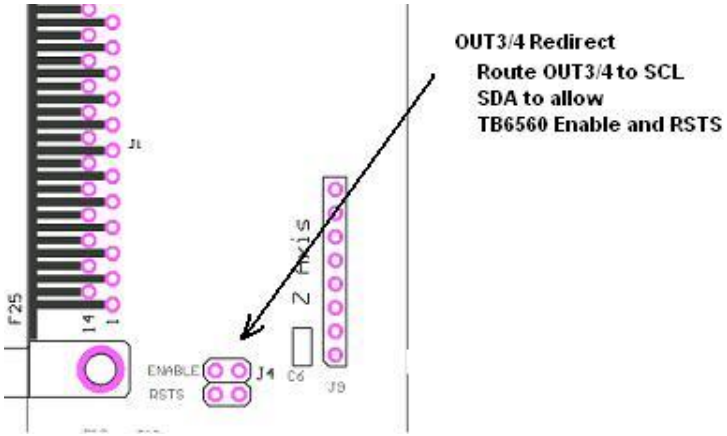
3.4 Stepper Driver Ports

GenY32 has four bipolar 3.5A/phase stepper motor driver output ports labeled X, Y, Z and A. The stepper motor driver is a TB6560. The step mode, torque and decay mode is set using a DIP switch for each TB6560 allowing individual settings for each axis. The onboard processor cannot read the DIP switch settings so any changes to these settings must be entered into the processors preference settings. When the parallel port is used the processor preference settings don't need to be changed.



3.5 TB6560 Reset and Enable Header

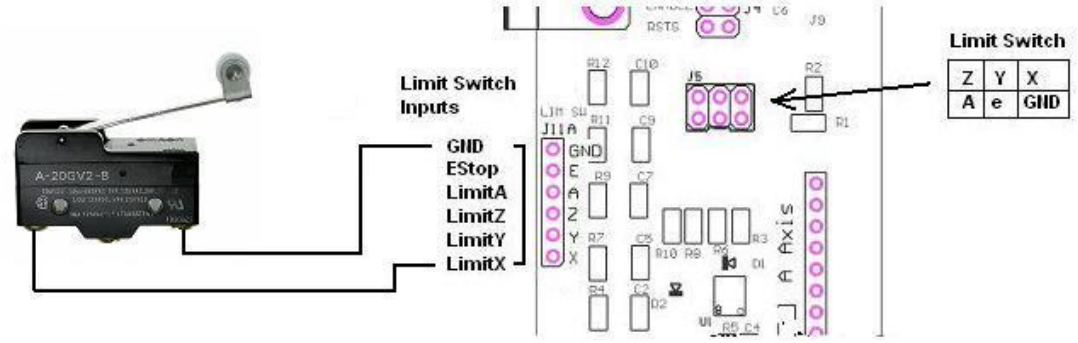
A four pin header allows OUT3 and OUT4 to be routed to the TB6560 Reset and Enable pins when shorting jumpers are installed. This allows host software to reset the state machine in each TB6560 to a known starting state and to disable or enable coil output drive MOSFETs. The Reset/Enable header is located close to the limit switch input area as shown below. By default the Enable inputs for each TB6560 is under control of the onboard processor which an enable each axis individually. When in Monitor Mode the processor can be set to control the Enable of all for axis depending on the level of OUT4.



3.6 Limit Switch Inputs

Limit switch inputs for the X, Y, Z and A axis are connected to the DB25 pins 11, 12, 13 and 15 respectively. The eStop switch input is connected to DB25 connector pin 10. The eStop and limit switch inputs are pulled high by a 10K ohm resistor and filtered by a 10uF capacitor to ground to reduce false triggers due to noise. When an external limit switch contact closes it pulls the limit switch input low the state of which is reflected on the DB25. eStop operates similarly to the limit switch input – closing the eStop with switch contacts pulls the eStop input low. Note that limit

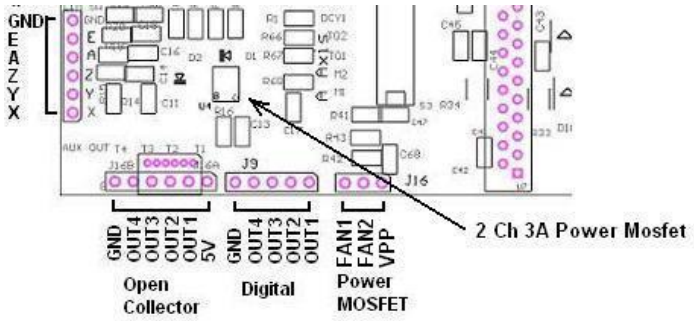
switches can be normally open or normally closed – the limit switch input will reflect the state accordingly – a normally open limit switch will indicate a high condition when not closed. Limit switch inputs are routed to two different functionally equivalent headers as shown below:



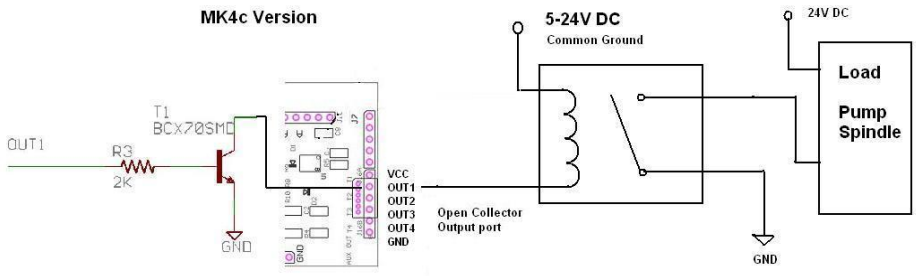
3.7

Au
and
on a

auxiliary output header. The transistor can be used to turn on a relay by attaching one side of the relay to the collector and the other side to +5, +12 or +24 volts. Each transistor is capable of driving about 100ma. There are four drive transistors on GenY32.



The open collector outputs can be connected as shown in the diagram below.

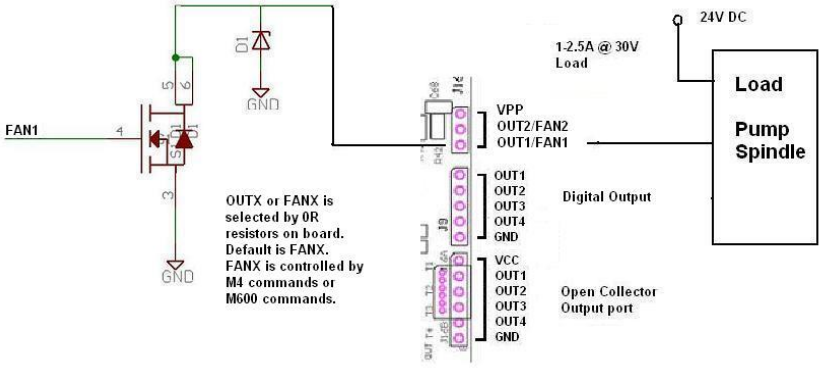


3.8 Digital Outputs

Auxiliary outputs OUT1, OUT2, OUT3 and OUT4 are connected to DB25 connector pins 14, 16, 17 and 1. Each OUT signal is directly connected to Auxiliary Digital output connector J9. A high output is 5V and a low output is 0V. The digital output port can be used to control external digital logic or very low current draw solid-state relays. Two additional motor control drivers can also be connector to digital out J9 – MM160 or MM166 drivers.

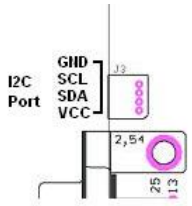
3.9 High Power MOSFET Outputs

GenY32 has two 3A 30V Power MOSFET outputs – these outputs can be used to drive high current peripherals such as spindles, pumps etc that require 24V 1-2A. No additional relay is required. The output of the Power MOSFET is connected as shown in the diagram below. The control of the Power Mosfet output is set using either an M4 or M600 commands (see G Code section). The M600 commands allow the controller to turn each Power Mosfet on and off with a resolution of 1msec. This allows the Power Mosfets to precisely control pumps and relays. The Power Mosfets are set using the FAN1 and FAN2 outputs.



3.10 I2C Port

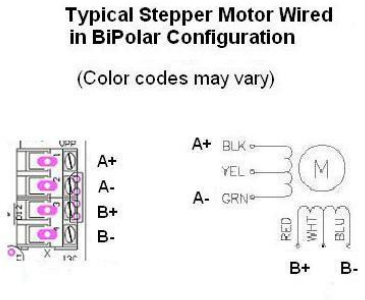
A small white four pin Molex connector located next to the DB25 connector is the I2C communications port. GenY32 processor can communicate with I2C devices as Master allowing the processor to control Smart Motor controllers such as the MM165, MM220 or sensors such as the 10DOF IMU6420 or IMU8420.



3.11 TB6560 Stepper Motor Driver

Motor Drive voltage should be between 10-37VDC. Average drive current (torque) is set using a DIP switch from 20% to 100%. The motor drive chip TB6560 has a 0.7ohm RDSon, which means that the chip requires forced air-cooling under most operating, conditions. The diagram below shows stepper motor connections for the SOC Robotics SM2006 and SM3006 stepper motors – wire color for other stepper motors may vary although most coil phase diagrams are similar. It is good practice to wire a fuse between the motor power supply and the TB6560. Do not turn motor power on if the logic side of the TB6560 is not powered – damage to the TB6560 driver chip may

result. If the motor will not turn then check your wiring. Never attach or detach stepper motor wires with power supply power turned on.



Adjusting Motor Drive Current

The TB6560 motor drive current is set by a DIP switch in four increments. Default setting is 3.5A – the maximum setting. It is possible to set the motor current in such a way that the control chip delivers too much current to the motor resulting in motor overheating. If the motor missteps at high step rates then the motor maybe undersized for your application and can not develop enough torque – in this case use a bigger motor. The stepper motor may also misstep if the rate of change of speed (acceleration) is too high exceeding the motors pull-in/pull-out torque. Increasing drive voltage will increase maximum step rate. The maximum step rate of a stepper motor is determined by drive voltage, coil inductance and step mode. Changing step mode from full step to half step can reduce effective drive torque by 40% so when selecting microstepping modes remember that effective torque is reduced accordingly and may result in miss stepping.

DIP Switch Settings

A DIP switch is used to set step mode (full, half, eight and sixteenth), torque (100%, 75%, 50%, 20%) and decay mode (0%, 25%, 50% and 100%). The diagram below shows the various settings, default setting is half step, 100% torque and no decay. When a DIP switch is set to "ON" the respective signal is pulled to ground. DIP switch changes should be made with power off so the correct selection is active at power up.

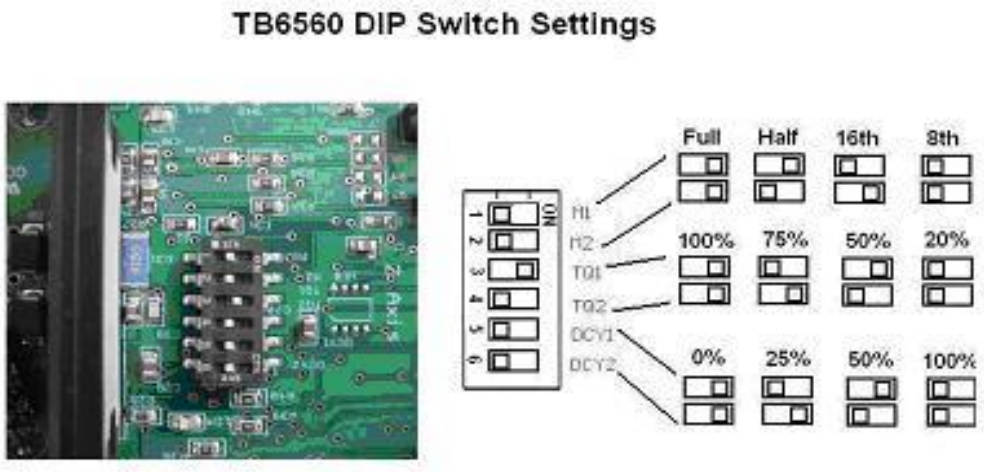


Figure 2-3. **DIP Switch setting and location.**

The TB6560 supports four different step modes set by DIP switch 1 and 2 shown in the diagram above.

The TB6560 has four torque modes. The maximum torque is 3.5A set by two 0.150 ohm sense resistors. Lower torque settings are selected by setting DIP switch 3 and 4.

The TB6560 supports four decay modes. Decay modes determine how fast residual current is discharged by the TB6560. As the TB6560 drives a stepping motor at faster step rates a back emf is produced by the rotation of the rotor. In order to discharge this back emf quickly fast decay is required. The TB6560 is shipped with no decay selected. According to the TB6560 data sheet the appropriate setting for the decay is best determined by observing the current signal on an oscilloscope. In the absence of an oscilloscope the best decay setting is determined experimentally. As a rule of thumb if you don't plan to run the stepper at its maximum rated step rate then leave decay at 0%.

Thermal Considerations

The TB6560 uses 0.150 ohm sense resistors – these resistors ensure the maximum current rating of 3.5A is reached. When using a stepper motor with a lower current rating set the torque DIP switches to a lower setting to prevent the TB6560 and stepper motor from overheating.

At the maximum torque setting 3.5A a 3.5A stepper motor will cause the TB6560 to overheat in a few minutes and shut down unless forced air-cooling is used. A RED led on the board turns on when the TB6560 enters thermal shutdown which occurs when the die temperature reaches 170deg C +- 20deg C and all outputs shutoff. The RED led goes out once the temperature falls to a safe operating level.

The TB6560 typically runs hot so forced air-cooling is mandatory under most operating conditions. For example, with a 3.0A NEMA23 stepper motor attached to the TB6560, no heat sink and no forced air the TB6560 will reach maximum operating temperature and shut down in just over two minutes. A small brushless DC 24V 0.1A 2.5" diameter fan blowing air across the TB6560 keeps the package at a constant operating temperature of 75C which is 60C less than without forced air cooling. Adding a heat sink lowers the operating temperature by another 15C to 60C.

Consult the factory for a recommend heat sink or the obtain specifications on the SOC heat sink kit for the TB6560.

Heat Sink Options

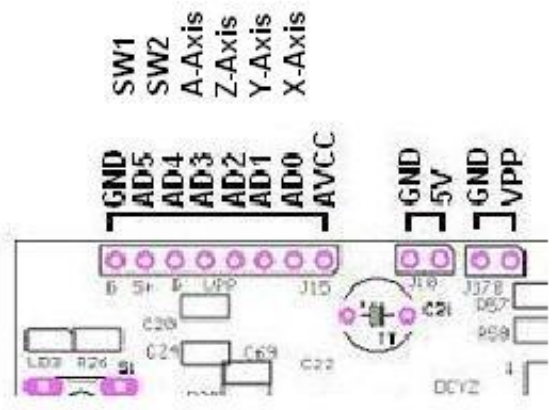
There are a number of hest sink options for GenY32. The HF4p is a solid one piece custom CNC'd Aluminum heat sink that attaches to all four TB6560's. The HF4p has mounting holes to allow up to three 40x40mm DC Fans to be mounted. Fan power is supplied by header J17B.

Transient Voltage Suppressor (TVS)

Three bidirectional TVS 600W diodes are installed across A+, A-, B+, B- and VPP, GND. These diodes conduct when the voltage across them exceeds 37 volts and have an extremely fast turn on time. They are designed to protect the driver chip from excessive inductive voltage spikes. The TVS diodes used on the TB6560 are made by Littlefuse part number P6SMB39CA.

3.12 Joystick/Analog Input Port

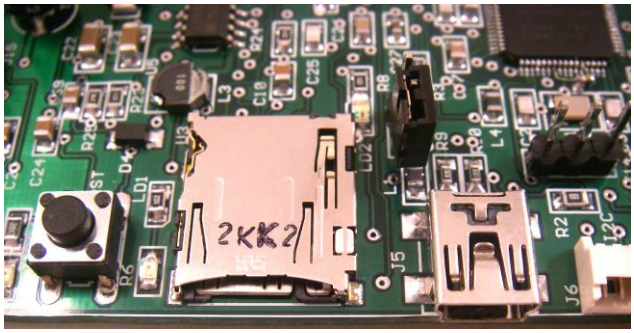
GenY32 has an analog input port (J15) configured either as digital output for Axis B and C Step and Direction output or as a four axis/two switch analog Joystick input port. By default J15 is configured as Axis B and C outputs. The AVR32 processor analog input section is designed to accept analog inputs ranging from 0V to 3V only even though the logic is running at 5V. Analog inputs must, therefore, not exceed 3V – a value above 3V produces incorrect analog input values. GStep automatically scans the analog input port at power up to determine if a Joystick device (CJ14 or CJ34A) is attached by seeing if all four analog inputs are sitting at approximately 1.5V – the resting position of the Joystick. By default Joystick mode is disabled on power-up and must be enabled by entering the GStep "j" command. GStep allows each axis to be enabled or disabled.



Analog inputs AD0-3 are mapped to the X, Y, Z and A axis respectively while AD4 and AD5 are considered switch inputs that control Joystick operation. The CJ14 and CJ34A Joystick peripherals are compatible with GenY32 Joystick port and can be directly connected. Note that the CJ34B Joystick is a I2C peripheral and attaches to the I2C port.

3.13 uSD Port

GenY32 has a uSD port supporting a FAT32 file system with storage capacity up to 32Gbytes. GStep automatically recognizes installed uSD cards on power-up. G Code files can be loaded onto the uSD card and executed using the 'fg filename' command. A list of uSD commands is below.



```
File command summary:
       fl - List files on uSD
       fs - Save system configuration parameters to file config.txt
       fp - Load system configuration parameters from file config.txt
       fsf filename - Save system configuration parameters to file filename
       fpf filename - Load system configuration parameters from file filename
       fe filename - Erase file filename
       fd filename - Dump contents of file filename
       fo filename - Start storing all following G Code to file filename
                     use fc command to close file and terminate storing
       fc - Close G Code storage file
       fg filename - Execute G Code file filename
       fi - SD card size
       fm - Initialize SD and mount
```

3.14 Smart Limit Switches

GenY32 supports Smart Limit Switches. LS12 and LS12R are two members of the SOC Robotics Smart Limit Switch family of sensors. The LS12 has an ATtiny45 processor and uses optical interrupt sensors while the LS12R has a more powerful ATxmega16E5 processor and uses optical reflective sensors. The processor pulls an output line low whenever any of the two optical sensors detect that pre-configured threshold conditions have been met such as proximity to an edge or object. The host controller (in this case GenY32) interrogates the Smart Sensor and retrieves trigger conditions. Mounting a Smart Sensor on the milling head allows the GStep controller to move to multiple home positions (in 2 dimensions) by placing reflective tape at know positions on the work surface.

A Smart Limit switch attaches to a limit switch input. Smart limit switches and regular limit switches can be mixed on the same limit switch input. Multiple Smart Limit switches can be used on the same limit switch input to perform different functions as each Smart Limit switch has a unique id. GStep is configured to automatically recognize Smart Limit switches on each limit switch input and configure it according to configuration options stored in a configuration file (if present).

4.0 Programming the AVR32 Processor

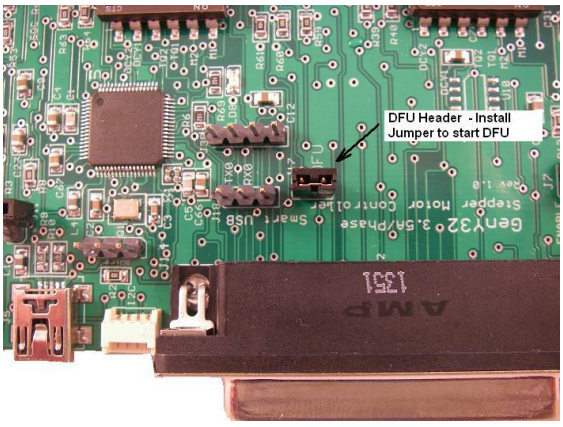
4.1 Overview

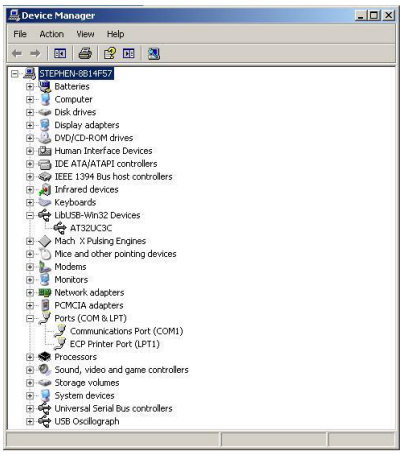
The AT32UC3C2512 can be programmed using the on chip DFU bootloader or the aWire port. **The preferred method of programming is to use the DFU bootloader**. Programming the processor using aWire will erase the DFU bootloader.

4.1 Programming the AT32UC3C2512

To program the AT32UC3C2512 using the DFU bootloader follow the steps below. The G Code application running on GenY32 board called GStep is updated on a regular basis. Current version is 2.05. The procedure below walks you through the process involved in re-flashing the latest version of GStep. GStep will be identified as GenY32_GStep_Controller_V2.05.hex.

- Download the Flip 3.4.7 programming utility from the Atmel web site
- Download the latest version of the GStep application from the SOC Robotics web site – this also includes the Mach3 Plugin
- Unzip the download go to the main Release folder
- If you haven't powered the board yet do the following quick check to make sure the board is communicating with the Windows desktop:
    - Power GenY32 and attach a USB cable from the board to a Windows PC and load the serial driver called "atmel_devices_cdc.inf" file located in the debug folder if this hasn't been done yet to install the CDC Virtual COM driver
    - Start a terminal app such as HyperTerminal and make sure GenY32 communicates with the desktop
- Install a two pin shorting jumper on the DFU header J17 (this connects processor pin PD21 to ground) and re-power the board – this starts the DFU bootloader. If this the first time the DFU bootloader has been started Windows will ask you to install a *.inf driver.
- Windows will ask for the location of a *.inf file for the DFU bootloader – it is located in the Flip utility subfolder called usb located under Atmel in Windows Program Files. If the DFU .inf file is properly installed it will show up in Device Manager as shown below.



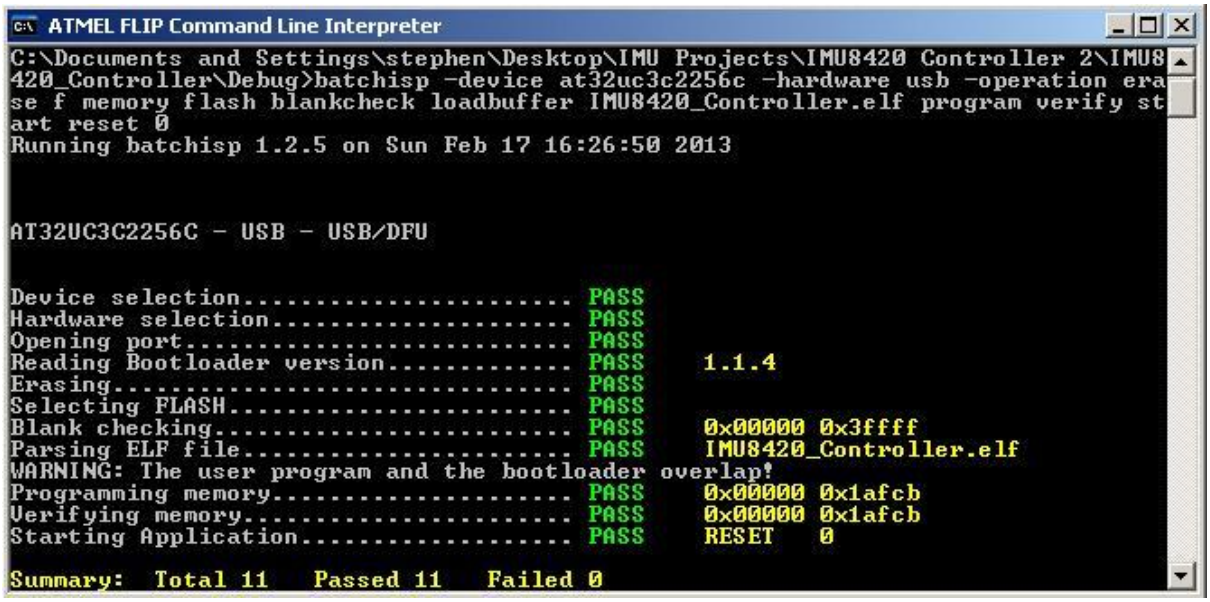


- Now run the batch file p512.bat located in the Release folder – this starts the programming process
- Contact SOC for the latest version of GStep for GenY32 or go to the download page at www.soc-robotics.com for the latest version.

The contents of p512.bat is below (p512.bat may contain a different hex file name):

```
batchisp -device at32uc3c2512c -hardware usb -operation erase f memory flash
blankcheck loadbuffer GenY32_Controller.hex program verify start reset 0
```

Running the p512.bat file should produce the results below.



DFU AT32UC3C2512 Bootloader

The AT32UC3C2512 has an on chip DFU bootloader that is activated by pulling signal PD21 on connector J17 to ground when the board is powered up. J17 is located close to the DB25 connector. The board is shipped with the DFU Bootloader pre-installed. If the Bootloader is erased it must be re-installed to use the programming procedure in the preceding section.

Once the bootloader starts it requests that a USB driver called "atmel_usb_dfu.inf" be installed, which is located in subfolder usb at "C:\Program Files\Atmel\Flip 3.4.7\usb". This folder is created when Atmel Programming Utility Flip is installed. Flip is available for download from the Atmel web site at www.atmel.com. Installing Flip also installs a batch processing application called batchisp.exe that will be used to reprogram the chips memory.

If the AT32UC3C2512 processor is reprogrammed using aWire the DFU bootloader is erased and must be reloaded. The procedure to reprogram the DFU bootloader is described below and in the file "Reloading GenY32 bootloader.txt" located in the Release folder of GenY32 GStep Release zip file available for down load from the SOC Robotics web site.

Re-Programming the AT32UC3C2512 DFU Bootloader

There are a few steps required to reload the AVR32 bootloader on GenY32's AT32UC3C2512 processor. Special programming hardware is required to reload the bootloader – it cannot be done via the USB interface.

The DRAGON programmer must be used in aWire mode along with avr32program.exe utility from AVR32 Studio 2.6. Any programmer capable of aWire or JTAG emulation can be used.

First connect the DRAGON to the aWire programming port on GenY32.

The programming operations documented below use commands that are part of the AVR32 Studio 2.6 distribution – download this tool chain from the Atmel web site. Note that a path to AVR32 Studio 2.6 must in the target folder. Execute the CMD command, move to the target folder and change the path as follows:

```
path C:\Program Files\Atmel\AVR Tools\AVR32
Studio\plugins\com.atmel.avr.toolchains.win32.x86_3.0.0.201009140852\os\win32\x
86\bin;C:\Program Files\Atmel\AVR Tools\AVR32
Studio\plugins\com.atmel.avr.utilities.win32.x86_3.0.0.201009140848\os\win32\x8
6\bin;%path%;
```

This points the current folder to the proper executables.

Now execute the following commands from the command line in sequence:

```
echo Erase the flash
  avr32program -p AVRDRAGON chiperase -F

echo Convert bootloader hex file to bin file
  avr32-objcopy -I ihex -O binary at32uc3c-isp-1.1.4.hex at32uc3c-isp-1.1.4.bin

echo Program the bootloader
  avr32program -p AVRDRAGON program -finternal@0x80000000 -cint -e -v -
O0x80000000 -Fbin at32uc3c-isp-1.1.4.bin

echo Program ISP configuration word
  avr32program -p AVRDRAGON program -finternal@0x80000000 -cint -e -v -
O0x808001F8 -Fbin at32uc3c-isp_cfg-1.1.4.dat

echo Program general purpose fuse bits
  avr32program -p AVRDRAGON writefuses -finternal@0x80000000 gp=0xF877FFFF

echo Reset MCU so it enters DFU - now you can use batchisp to load the
application.
  avr32program -p AVRDRAGON run -R
```

The commands above are in the file: program_at32uc3c-isp-1.1.4b.sh

For some reason this file does not run correctly when invoked so you need to enter the commands manually from the command line. You can do this by cutting from the text file and pasting into the command line prompt. You can ignore the echo commands.

To re-start the bootloader connect PD21 to ground and repower GenY32.

aWire Emulation Port

The aWire port is a three wire (GND,VDD,RESET) programming port that several Atmel programmers can use to re-flash the AVR32. A cost effective programmer is Atmel's Dragon programmer. The Atmel web site has additional information on the Dragon.

5.0 GStep G Code Command Description

5.1 Overview

Please note that GStep V2.09 has many new updates and features, which are not completely documented in this manual yet but should be shortly. GStep now supports six axis by default. The Step and Direction signals for Axis B and C are assigned to Port J17 pins 6,7,4 and 5.

GenY32 has a dedicated processor that interprets and executes G Code commands sent to it over the USB Serial Interface. G Code is a language introduced many years ago to provide a common protocol for the control of CNC equipment. The G Code standard is described by RS274NGC. This manual does not intend to teach G Code or describe in detail how G Code is used but rather to describe the specific implementation in GenY32. The version of G Code implemented in GenY32 upgraded frequently. For the latest version release and change log go to GenY32 product page under the download section of the SOC Robotics web site. The G Code program executing on GenY32 is called GStep. The current version of GStep is output in the sign on message when GenY32 powers up or the "?s" command is entered.

GenY32 supports several G, M, S, T and related NS274 commands along with proprietary commands unique to GenY32. Commands are entered one after the other at the dollar sign prompt. Commands can be separated by space or not and are usually case insensitive except for a few non-G Code commands such as the File Subsystem commands. A command is executed after a Carriage Return <CR> is entered. All commands are echo'd automatically. At the completion of a command the prompt character is output indicating GenY32 is ready for the next command. Host software can use reception of the prompt character to manage the flow of commands. Some commands take much longer than others to execute so a desktop application is required to send commands to GenY32 at a rate that does not overflow the receive buffer.

GenY32 supports a number of proprietary commands to allow interaction with attached peripherals such as a joystick or LED display. The host can request information from GenY32 using various status commands such as Get Current Position or Get Joystick Position. In lead screw CNC applications Backlash Compensation can be entered on a per axis basis in integer steps – backlash correction are applied symmetrically for all direction changes.

The en axis position changes caused by joystick movements to be accurately tracked so that subsequent G Code position commands can move to the new correct position while maintaining correct absolute position. G0,1,2,3, I,J,K and F commands can be entered as integers or as single precision floating point numbers.

The X, Y and Z axis are assumed to be linear. The fourth axis is interpreted as either a linear or rotary axis using the A or B command respectively. Parameters after the B command are interpreted as angles in degrees. The relationship between an axis parameter and the number of steps executed by the stepper is set by a resolution parameter. The default resolution is 0.050" per 360 rotation of the stepper or 0.00025" per step– so a command such as N0001 G1 X0.250 F4.25 causes the X axis stepper to rotate 360 degrees five times or to execute 5*200 = 1000 steps at a position change rate of 4.25 inch per minute. The resolution parameter is programmable. The current release assumes the resolution parameter once set is the same for all axis except for rotary motion – this parameter is set separately. In the examples <CR> indicates a carriage return and <LF> indicates a line feed.

The Table 5-1 below summaries the command codes supported by GenY32.

Table 5-1 G Code Commands Supported By GenY32

Command	Function
G0	Rapid Move maximum rate on each axis
G1/G01	Linear Move – constant speed
G2/G02	Circular Interpolation - Clockwise
G3/G03	Circular Interpolation – Counter Clockwise
G4/G04 Psss	Dwell in seconds
G20	Select imperial units
G21	Select metric units
G46Cxxx	Back lash Compensation – C is X,Y,Z or A
G90	Absolute Positioning
G91	Relative Positioning
G92	Set home position
G94	Output status
G96mmmmm	Dwell for mmmmm mseconds
G97	Output current axis position
I,J,K,R	Variables for Circular Interpolation
N	G Code number
F	Feed Rate – Fxx.xxxx ipm or mmpm
X	X Axis (linear or rotary)
Y	Y Axis (linear or rotary)
Z	Z Axis (linear or rotary)
A	A Axis (linear or rotary)
B	B Axis (linear or rotary) (requires MD288)
C	C Axis (linear or rotary) (requires MD288)
M41|2	Aux Output 1 On/Off
M43|4	Aux Output 2 On/Off
M45|6	Aux Output 3 On/Off
M47|8	Aux Output 4 On/Off
H	Home machine position menu
J	Enable/disable Joystick recognition
CM	Set Monitor Mode
CC	Set Control Mode
SP	Output current position
SB	Output Backlash Values
SJ	Output Joystick Values
SA	Output Aux Port Values
SX	Output Step Mode Values
SE	Enter preference setup menu
LE	Leave preference setup menu
SS	Output Status
P	Display preference settings
?	Help menus ??,?g,?m, etc
!	Enter Test Mode

5.2 G Code Command Description

Supported G Code commands are described in detail in this section.

G0/00 Rapid Move - Maximum rate on each axis

Rapidly move to the new position moving each axis at the maximum specified rate.

Example: N00001 X1.0 Y1.0 Z0.25 A2.25 F8.0 <CR> - Move all four steppers at the maximum step rate until the destination is reached.

G1/01 Linear Interpolation – Specified feed rate

Move the X, Y, Z and A axis at the specified feed rate so all axis arrive at the destination coordinates at the same time. GenY32 is capable of 100,000 steps per second when executing a linear interpolation move.

Example: N00001 X1.0 Y1.0 Z0.25 A2.25 F8.0 <CR> - Move all four steppers at the different step rates so they all arrive at their respective destinations at the same time.

G2/02 Circular Interpolation - Clockwise

Move the XY, XZ or YZ axis in an clockwise arc from the current position to the destination position using parameters defined by the I,J or R settings. If a new Z or A position is present then movement on these axes will also occur. GenY32 supports both center and radius circular interpolation. Helical interpolation is supported for all other axis.

Example 1: N00001 X1.0 Y1.0 I0.0 J0.0 <CR> - Assuming current position is (1.0, 0.0) move in an arc to (1.0, 1.0) using (0.0,0.0) as the origin.
Example 2: N00001 X1.0 Y1.0 R1.0 <CR> - Assuming current position is (1.0, 0.0) move in an arc to (1.0, 1.0) using a radius of 1.0.

G3/03 Circular Interpolation – Counter Clockwise

Move the XY, XZ or YZ axis in an counter clockwise arc from the current position to the destination position using parameters defined by the I,J or R settings. If a new Z or A position is present then movement on these axis will also occur. GenY32 supports both center and radius circular interpolation. Helical interpolation is supported for all other axis.

Example 1: N00001 X1.0 Y1.0 I0.0 J0.0 <CR> - Assuming current position is (1.0, 0.0) move in an arc to (1.0, 1.0) using (0.0,0.0) as the origin.
Example 2: N00001 X1.0 Y1.0 R1.0 <CR> - Assuming current position is (1.0, 0.0) move in an arc to (1.0, 1.0) using a radius of 1.0.

G4/G04 Psss Dwell in seconds

Set dwell time in seconds. GenY32 pauses for the specified delay period then return the dash prompt.

Example: N00001 G4 P10 <CR> - Delay for 10 seconds.

G20 Set Imperial Units

G21 Set Metric Units

G90 Absolute Positioning

Set absolute positioning mode.

G91 Relative Positioning
Set relative positioning mode. Note relative position mode is not implemented but is scheduled for implementation in new version.

G92 Set home position
Zero all four axis to 0.0000. Defines a new absolute start position.

G94 Output status
Output the system status such as current position, backlash setting, etc.

G96mmm Dwell for mmm mseconds
Delay for a specified period of time measured in mseconds. Alternative command to G4 for finer resolution delays.

Example: N00001 G9612 <CR> - Wait for 12mseconds before accepting the next command.

G97 Output current axis position
Output the current position of all four axis. The output is in floating point format using the current resolution in effect. The command converts the current step position of each axis in 32 bit Signed Long format to floating point format using the current axis resolution.

5.3 M Command Description
Supported M Code commands are described in detail in this section. The M code commands are used to control the four Auxiliary digital output signals AUX1 to AUX4, FAN1 and FAN2 outputs.

M3/M03 - Spindle Output On
M3 or M03 turns the Spindle (Aux 1 output) on. The Aux output port is an NPN open collector circuit. A high level pulls the line to ground while a low level turns the output off. M5 turns the Aux 1 output off.

M5/M05 - Spindle Output Off
M5 or M05 turns the Spindle (Aux 1 output) off. The Aux output port is an NPN open collector circuit. A high level pulls the line to ground while a low level turns the output off. M3 turns the Aux 1 output on.

M7/M07 - Pump Output On
M7 or M07 turns the Pump (Aux 2 output) on. The Aux output port is an NPN open collector circuit. A high level pulls the line to ground while a low level turns the output off. M9 turns the Aux 1 output off.

M9/M09 – Pump Output Off
M9 or N09 turns the Pump (Aux 2 output) on. The Aux output port is an NPN open collector circuit. A high level pulls the line to ground while a low level turns the output off. M7 turns the Aux 1 output off.

M41|2 - Aux Output 1 On/Off
M41 turns the Aux 1 output on – high level on Relay Expansion port pins and a logic low for the Aux output port. The Aux output port is an NPN open collector circuit. A high level pulls the line to ground while a low level turns the output off. M42 turns the Aux 1 output off.

M43|4 - Aux Output 2 On/Off
M43 turns the Aux 2 output on – high level on Relay Expansion port pins and a logic low for the Aux output port. The Aux output port is an NPN open collector circuit. A high level pulls the line to ground while a low level turns the output off. M44 turns the Aux 1 output off.

M45|6 - Aux Output 3 On/Off
M45 turns the Aux 3 output on – high level on Relay Expansion port pins and a logic low for the Aux output port. The Aux output port is an NPN open collector circuit. A high level pulls the line to ground while a low level turns the output off. M46 turns the Aux 3 output off.

M47|8 - Aux Output 4 On/Off
M47 turns the Aux 4 output on – high level on Relay Expansion port pins and a logic low for the Aux output port. The Aux output port is an NPN open collector circuit. A high level pulls the line to ground while a low level turns the output off. M48 turns the Aux 4 output off.

M49|a - FAN1 Output On/Off
M49 turns the FAN1 output on. The FAN1 output port is a 3.5A Power Mosfet. A high level pulls the line to ground while a low level turns the output off. M4a turns the FAN1 output off.

M4b|c - FAN2 Output On/Off
M4b turns the FAN2 output on. The FAN2 output port is a 3.5A Power Mosfet. A high level pulls the line to ground while a low level turns the output off. M4c turns the FAN2 output off.

M4d|e - OUT1-4, FAN1-2 All On/Off
M4d turns OUT1-4 and FAN1-2 outputs on. M4e turns all the outputs off.

M601 - FAN1 PWM Mode Setup
The M601 command turns control of the FAN1 output over to the 1 msecond timer interrupt handler. This allows the controller to turn the FAN1 Power Mosfet on and off with msecond precision. Fluid pumps are typically tuned on for short periods and left off for long time periods. For example a typical on time may be 15msec with an off time of 5 seconds. The on time is entered as a floating point number in seconds as is the off time (period). Syntax is given below:

M601[i,a,h]ox.xxxpy.yyyy - where mode is either i- timer interrupt, a- analog, h-hardware PWM – note analog and hardware PWM modes not supported in this version.

- o on time in seconds x.xxx
- p period in seconds – period must be greater than on time

M601io0.015p5.00 - On pulse of 15msec with a period of 5 seconds
M601io0.30p2.34 - On pulse of 0.3 seconds with period of 2.34 seconds

M602 - FAN1 PWM Mode Off
M602 turns FAN1 PWM mode off and releases control for FAN1 On/Off back to M49/a command control.

M603 - FAN2 PWM Mode Setup

The M603 command turns control of the FAN3 output over to the 1 msecond timer interrupt handler. This allows the controller to turn the FAN2 Power Mosfet on and off with msecond precision. Fluid pumps are typically tuned on for short time duration and left off for long time periods. For example, a typical on time may be 15msec with an off time of 5 seconds. The on time is entered as a floating point number in seconds as is the off time (period). Syntax is given below:

M603[i,a,h]ox.xxxfy.yyyy - where mode is either i- timer interrupt, a- analog, h-hardware PWM – note analog and hardware PWM modes not supported in this version.

- o on time in seconds x.xxx
- p period in seconds – period must be greater than on time

M601io0.015p5.00 - On pulse of 15msec with a period of 5 seconds
M601io0.30p2.34 - On pulse of 0.3 seconds with period of 2.34 seconds

M604 - FAN2 PWM Mode Off

M604 turns FAN2 PWM mode off and releases control for FAN2 On/Off back to M4b/c command control.

5.4 Other RS274 Commands

Other supported RS274 commands are described in detail in this section.

N G Code Line Number

The Line number is optional and is ignored but helpful to read G Code files.
Example: N0001 G0 X2.025 Y3.125 F4.000 <CR>

F Feed Rate – Fxx.xxxx ipm or mmpm

The Feed Rate is used by G1, G2 and G3 commands to set stepper step rate. Step rate is related to step resolution that is related to the specific mechanical configuration of an application. The feed rate remains in effect until changed.

Example: N00001 G1 X2.000 Y2.0000 F4.0000 <CR>

X X Axis

Sets a new position for the X axis. All X Axis moves are linear. The number of steps and direction moved depends on the difference between the current X axis position, the end position, direction change (if any), backlash compensation and resolution. For example if the current X position is 1.00000 the target X position is 1.25000, step resolution is 0.00025 then the stepper steps 10000 times clockwise.

Examples: N0001 X0.2500 F4.0 <CR> - Move to X=0.2500 at 4.0ipm
N0001 X2 F3 <CR> - Move to X=2.0000 at 3.0ipm

Y Y Axis

Sets a new coordinate for the Y axis. All Y Axis moves are linear.

Z Z Axis
Sets a new coordinate for the Z axis. All Z Axis moves are linear.

A A Axis
Sets a new coordinate for the A axis. All A Axis moves are linear.

B B Axis
Sets a new coordinate for the B axis. All B Axis moves are rotary. The B Axis is actually the fourth axis re-interpreted as rotary motion. The B Axis provides a means to control rotary motion such as a turntable. Parameter is degrees.

B B Axis
Sets a new coordinate for the B axis. All B Axis moves are rotary. The B Axis is actually the fourth axis re-interpreted as rotary motion. The B Axis provides a means to control rotary motion such as a turntable. Parameter is degrees.

Example: N00001 G1 X1.0 B25.0 F4.0 <CR> - Move to X Axis location 1.0 and rotate B axis 25.0 degrees.

5.5 Proprietary Commands
GenY32 implements proprietary commands not supported by the RS274 command set. Commands to set axis resolution, return current position, set backlash compensation, etc are members of this set of commands.

? Output online Help
Output a summary of all commands by entering the H command.

If the "h" command is entered then the following help message is output. Note that the actual help may have new features added at each revision upgrade.

```
-h
MC433 Stepper Controller - Gstep V1.63
Copyright 2006, SOC Robotics, Inc.
Type H for help menu - Control mode active

-
Single Step commands:
 > = step X CW
 < = step X CCW
 i = step Y CW
 o = step Y CCW
 u = step Z CW
 d = step Z CCW
 z = step A axis CW
 w = step A axis CCW

G Code Commands ([]-optional - floats  xx.xxxxx):
 Fourth axis is linear A or rotary B (angle)
 G0 Xx.xx Yx.xx Zz.zz Aa.aa Bb.bbb- Rapid move
```

```
 G1 Xx.xx Yx.xx Zz.zz Aa.aa Bb.bbb Ff.fff - Linear Interpolation
 G2 Xx.xx Yx.xx Zz.zz [Ii.ii Jj.jj|Rr.rrr|Daaa.a] - Circular Interpolation - CW
 G3 Xx.xx Yx.xx Zz.zz [Ii.ii Jj.jj|Rr.rrr|Daaa.a] - Circular Interpolation -
CCW

 G46Cxxx - Set backlash for each axis where C is X,Y,Z or A
 G4Ppppp - Dwell for pppp seconds
 G92  - Reset origin x,y,z,a
 G96mmmm - Dwell mmmm msec
 G97  - Output Position

M Code Commands:
 M5[1|2] - Aux 1 1-ON 2-OFF
 M5[3|4] - Aux 2 1-ON 2-OFF
 M5[5|6] - Aux 3 1-ON 2-OFF
 M5[7|8] - Aux 4 1-ON 2-OFF

Other Commands:
 sp - Get Current position
 ss - Get All Status information
 sb - Get Backlash values
 sj - Get Joystick values
 sl - Get Limit switch values
 sa - Get Aux IO values
 sd - Get Digital IO values
 sx - Get Step mode values
 se - Enter preference setting mode
 le - Leave preference setting mode
 cc - Set command mode active
 cm - Set monitor mode active
 p  - Display preference settings
 b  - Change baud rate
-
```

Single Step Commands

GenY32 is capable of single stepping the steppers using simple character sequences entered at the command prompt. Single step commands are executed immediately and do not require a <CR> or <LF>. The commands are summarized in the table below and are case insensitive:

Step Command	Action
<	Step X axis CW
>	Step X axis CCW
i	Step Y axis CW
o	Step Y axis CCW
d	Step Z axis CW
u	Step Z axis CCW
z	Step A axis CW
w	Step A axis CCW

CM Set Monitor Mode

Set Control Monitor mode – the G Code Processor releases all step/direction and Aux IO lines and allows the parallel port to take control. The Control Monitor mode can be used if step/direction control signals are sent to GenY32 via the parallel port. The Control Monitor Mode must be set to allow Flash Programming of the Motor Controllers.

CC Set Control Mode

Set Control Active Mode – the G Code processor takes control of the Step/Dir signals. The parallel port cable should not be connected in this state or signals sent to the Motor Controllers by the G Code Processor will conflict.

SP Output Current Position

Output the current position of all four axis. The output is in floating point format converted using the current resolution in effect. GenY32 maintains current position using a 32bit Signed Long.

SB Output Backlash Setting

Output the current backlash settings for all four axis - X, Y, Z and A. Backlash settings are assumed symmetric and are stated in steps.

SJ Output Joystick Values

Output the current joystick values for all four axis - X, Y, Z and A and the JSW switch level. The joystick is an 8 bit number converted to plus or minus 128. The switch is either open or closed.

SA Output Auxiliary Output Port Settings

The state of the four auxiliary output signals is returned.

SS Output Status Summary

Output all the current status information of GenY32.

! Enter Test Mode

Enter test mode by entering the '!' character. In test mode GenY32 drives one or all Motor Controllers at a user entered step rate. This helps setup the steppers during initial installation. The test mode command is as follows:

```
-!
Test Mode (Enter 'e' to exit)
Commands:
  <axis>[c|w]<step rate/sec>
  Example:  Xc200 Yw1500 Zc700 <CR>
>
```

The following example starts the x and y axis stepping at 200 pps (pulse per second – steps).

```
>xc200 yw200<CR>
```

SE Enter Preference Configuration Menu

This command is used to enter the Preference Configuration Menu. The G Code processor assumes that the resolution of each axis is 0.00025in per step, there is no backlash, maximum rapid move is 20ipm and step direction polarity is P N N N. Entering the "se" command enters the Preference setting menu:

```
-se
Entering Preference Mode
Type 'le' to leave - dp to display preferences - ss save settings
=
```

For see the current preference settings enter "dp".

```
=dp
Preference Settings:
Baud rate = 115200
Max Feed rate (mf)= 20.0
Resolution [X,Y,Z,A,B,C] (rr)= 0.000125 0.000125 0.000125 0.000125 0.002 0.002
Backlash [X,Y,Z,A] (bl)= 0 0 0 0 0 0
Direction polarity [X,Y,Z,A] (pl)= P N N N N N
=
```

Now change any preference setting by entering a two character preference selection command followed by the new values. For example to change the resolution from 0.00025 to 0.000125 enter the following command:

```
=rr0.000125 0.000125 0.000125 0.000125 0.000125 0.000125
=
```

Changed preferences are not saved in EEPROM until the "ss" command is entered.

Preference setting changes take effect immediately for the time GenY32 is powered but revert to stored settings unless saved by the "ss" command. The "le" command is used to leave Preference Configuration Mode.

5.6 Updating Flash Contents

GenY32 preference settings are stored in the G Code processors on chip Flash. Flash contents can be changed using the Change Preference command. Controller configuration settings can also be stored and load from uSD – see the uSD command section below for more information.

5.7 I2C Pass Through Protocol

GenY32 has an I2C communications port that allows GStep to communicate with and control a number of smart peripherals such as LED displays, Joysticks and smart relay boards. SOC Robotics has a rich set of smart I2C peripherals all of which can be controlled by GenY32 using a unique pass through protocol that allows the Host PC to send I2C commands to GenY32 that are "passed through" to the I2C interface – this ensures GenY32 is able to accommodate new I2C peripherals as they become available without waiting for new version of GStep. The Pass Through protocol will be implemented in the next release of GStep.

5.8 uSD Commands

The on board uSD can store configuration parameters, G Code files and error log information. The following commands are supported:

```
File command summary:
       fl - List files on uSD
       fs - Save system configuration parameters to file config.txt
       fp - Load system configuration parameters from file config.txt
       fsf filename - Save system configuration parameters to file filename
       fpf filename - Load system configuration parameters from file filename
       fe filename - Erase file filename
       fd filename - Dump contents of file filename
       fo filename - Start storing all following G Code to file filename
                     use fc command to close file and terminate storing
       fc - Close G Code storage file
       fg filename - Execute G Code file filename
```

```
fi - SD card size
fm - Initialize SD and mount
```

The fs and fp commands store and load configuration parameters from a file called config.txt. Config.txt is a plain text file the first lines of which contain a brief description of the file. After the first two lines parameters are stored with a leading parameter label followed by the label value. Some parameters, such as step resolution, require six values – one for each axis. The fs commands saves all current configuration parameters to config.txt. Examining the contents of the file identifies all the possible configuration parameters used by the controller. There is no particular order nor need to include all parameters in a configuration file. If a parameter is not included system defaults are that last setting are used.

To save or retrieve settings from another file use the fsf and fpf commands.

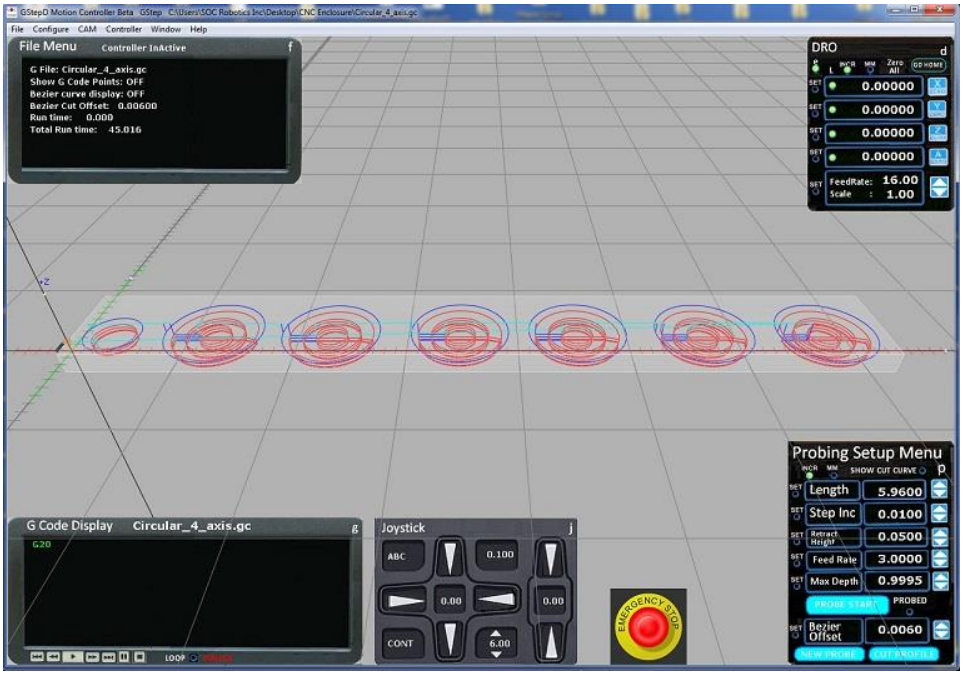
Delete a file using the fe command. The list the contents of a file use the fd command.

To store a sequence of G Code commands use the fo and fc commands. fo filename starts recording all following G Code commands to file filename until an fc command is entered. G Code commands are executed as they are entered. Note that this process can not be used to enter plain text to a file.

To execute G Code stored in a file use the fg filename command.

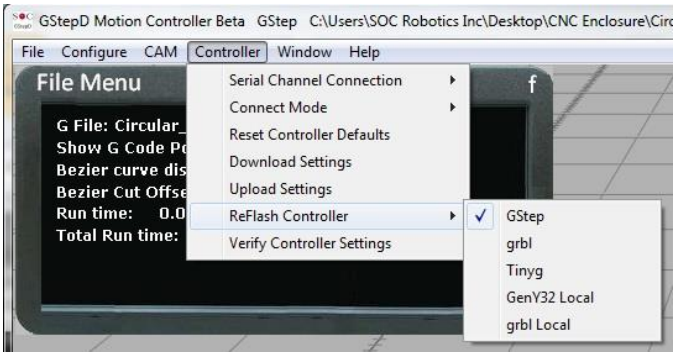
5.9 Windows Desktop GUI GStepD

A new Windows GUI called GStepD is in development and will be released shortly. GStepD is a comprehensive desktop G Code Sender, Controller configuration and G Code Path visualization tool. GStepD also supports rudimentary 2.5D CAM functions. GStepD provides complete controller configuration and g code interpreter download control with support for GStep, grbl and tinyg built-in. GStepD runs on XP, Win7,8 and 10. Beta release is scheduled for May 2016.



GStepD includes a

controller update feature that checks if newer versions and updates the controller automatically. Manual reflashing is no longer necessary.

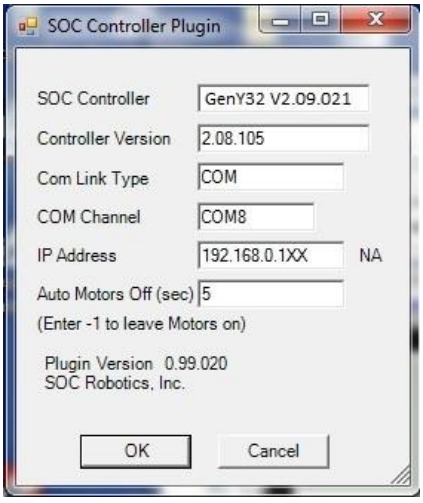


5.10 Mach3 Plugin Support

GStep includes Mach3 plugin support. If you want to use Mach3 to control GenY32 download GenY32 GStep V2.09.02/Mach3 V0.99.02 installer from the download page at www.soc-robotics.com. This installer is for Win7, 8 and 10. If running on XP download GenY32 GStep V2.09/Mach3 V0.99 Installer (for XP).

Running the installer loads a plugin into Mach3's plugin folder, installs a serial driver into the C:\Windows folder and creates a master folder C:\SOCRobotics with several subfolders containing drivers, documentation, different hex files for GStep and programming utilities.

The latest release supports six axis operation, individual axis enable/disable and slaving of X,Y or Z to A,B or C. An automatic motor shutoff feature is available on the plugin panel.



5.11 grbl V0.8c G Code Interpreter Support

grbl v0.8c has been ported to GenY32. There are number of grbl versions available in the C:\SOCRobotics\GenY32 folder. The basic grbl code has been enhanced with support for tandem drive (A axis locked to X axis) and four axis operation. Complete source for each of the ports is available on the download page GenY32 GRBL V1.02 4 Axis Source AVR Studio 6.2

Project. A port of grbl V0.9 is underway and should become available in May 2016. Grbl was ported using Atmel's AVR Studio 6.2 IDE.

5.12 tinyg G Code Interpreter Support

A tinyg G Code interpreter is being ported to GenY32 and will be released later in May 2016. tinyg is an experimental port that support jerk control, json protocol support and six axis control. Once the port is complete project source code will be released. tinyg was ported using Atmel's AVR Studio 6.2 IDE.

6.0 Electrical and Mechanical Description

6.1 Electrical Specifications

Electrical

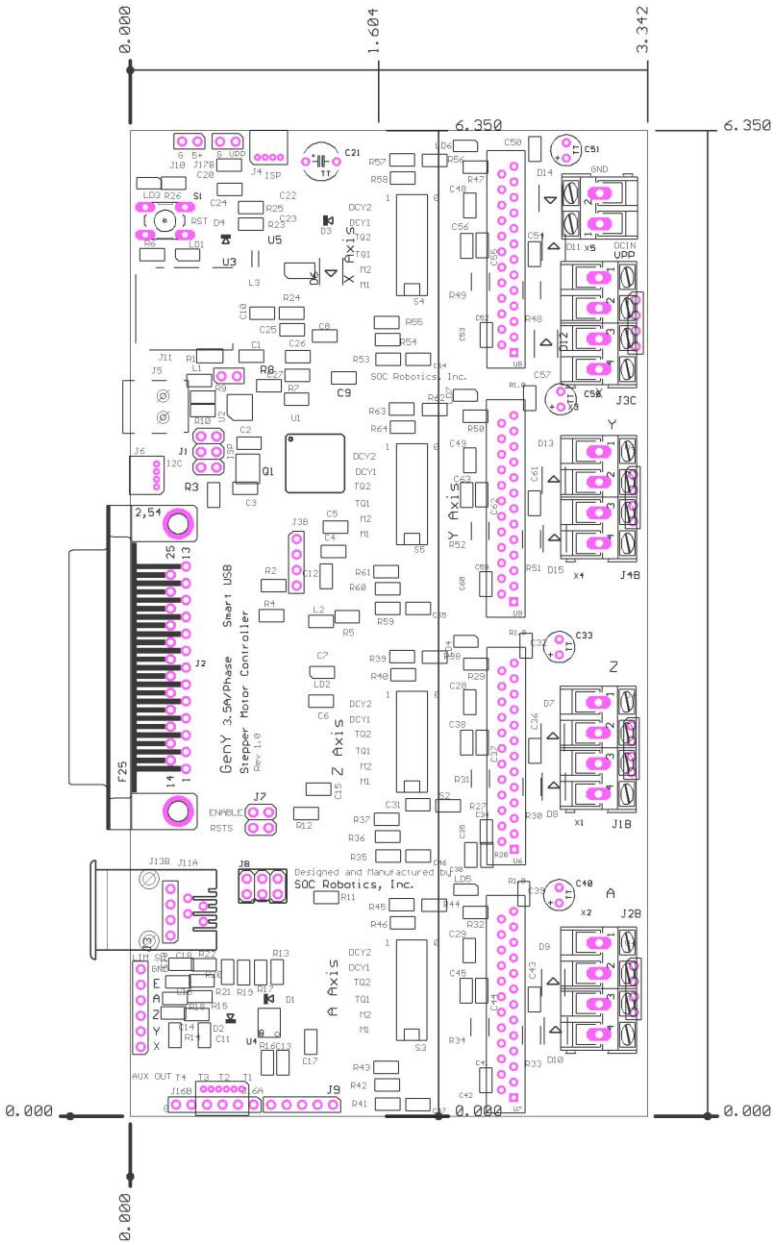
Input power: 12-32VDC @ 160ma noise protection plus reverse polarity protection

Mechanical

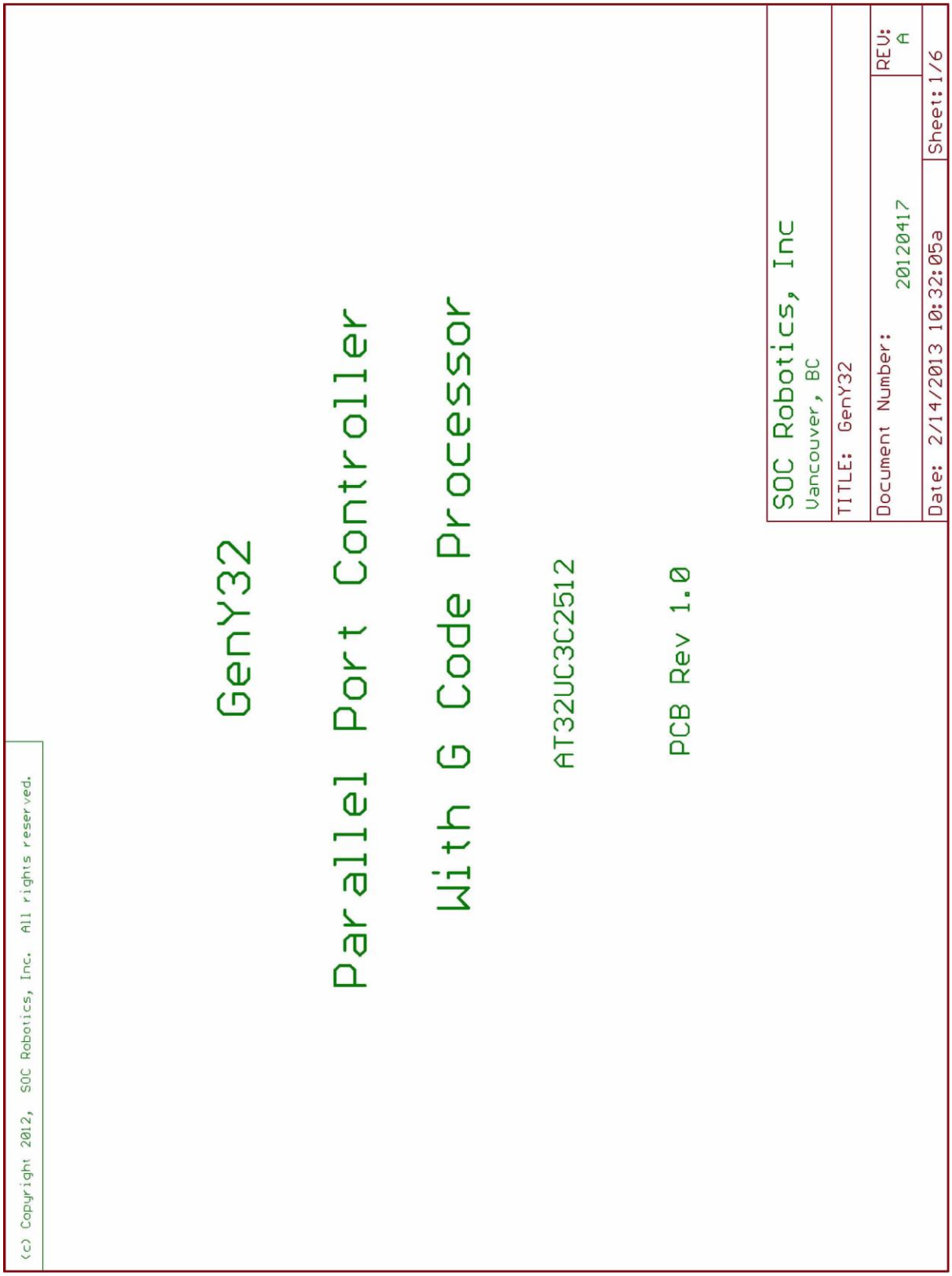
Dimensions: 3.35x6.35 in
Weight: 40 grams

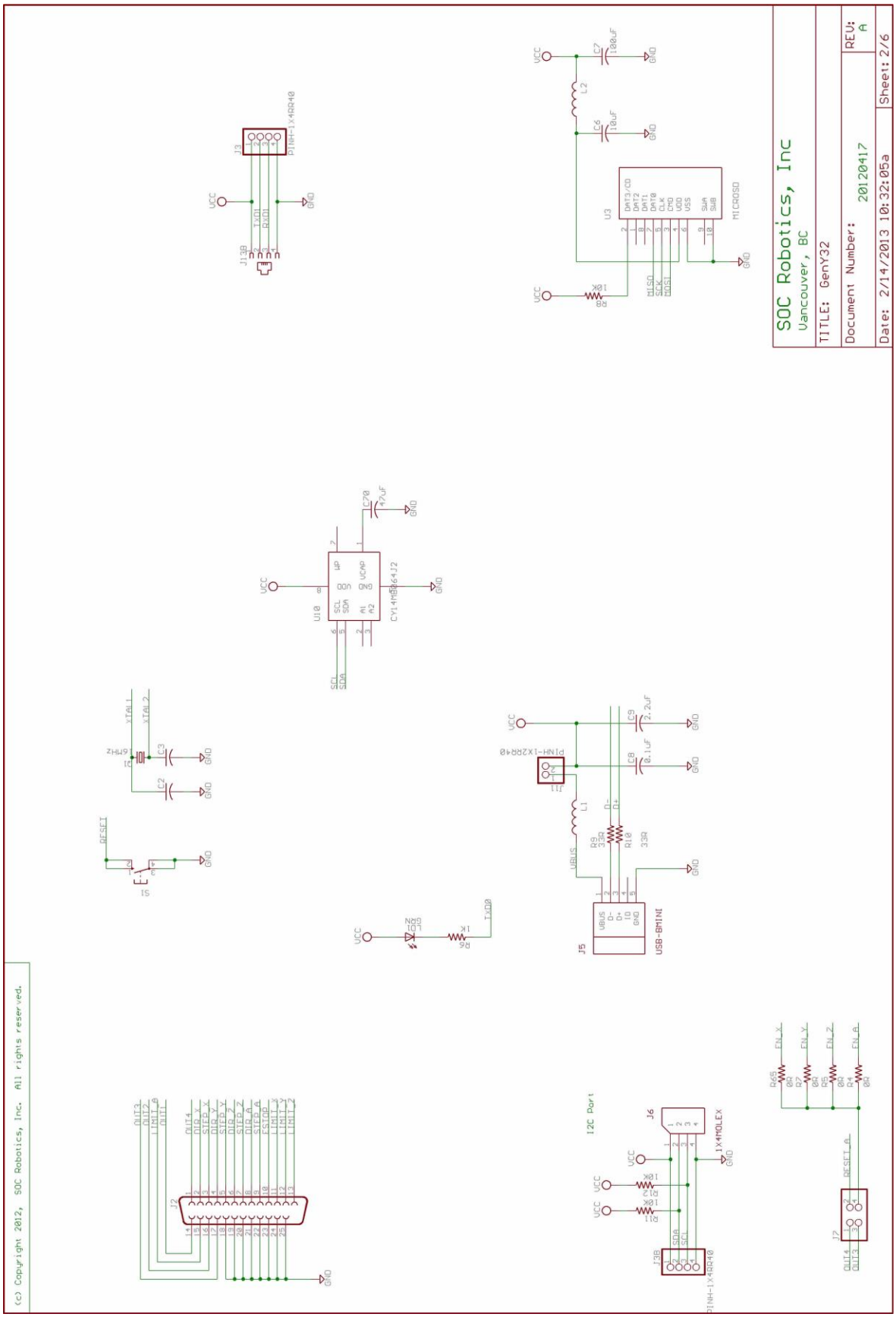
6.2 Mechanical Dimensions

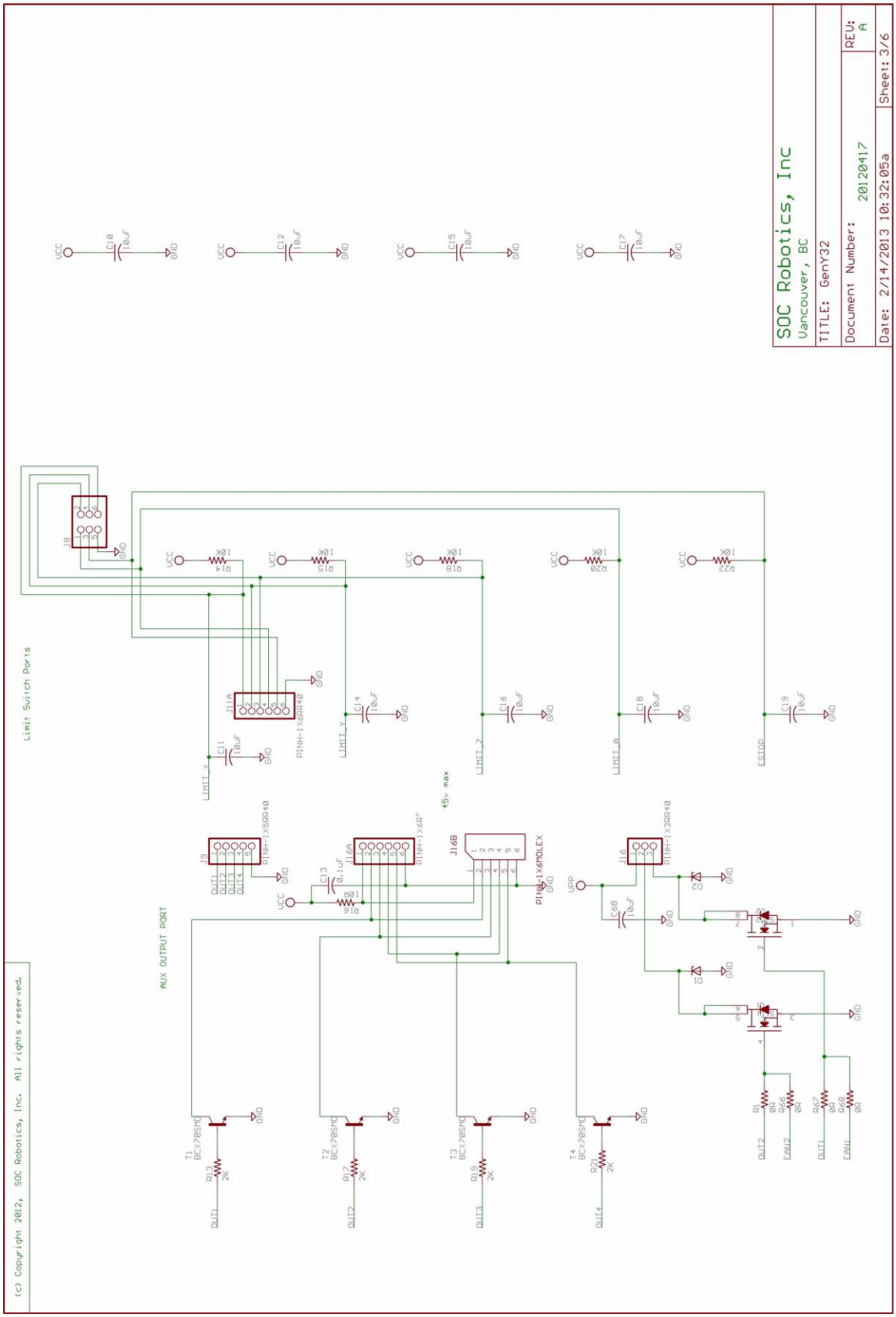
Board dimensions are stated in inches.

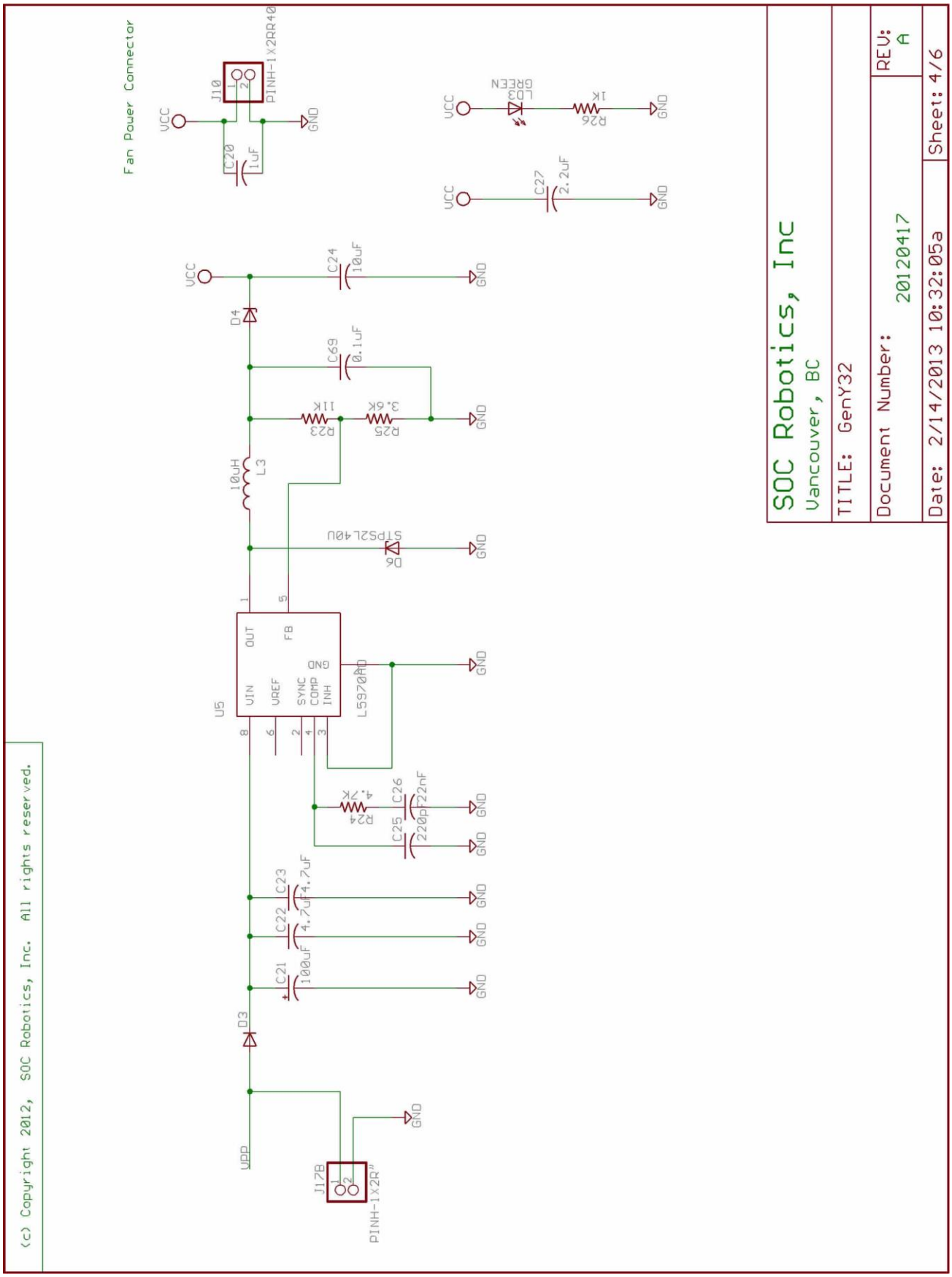


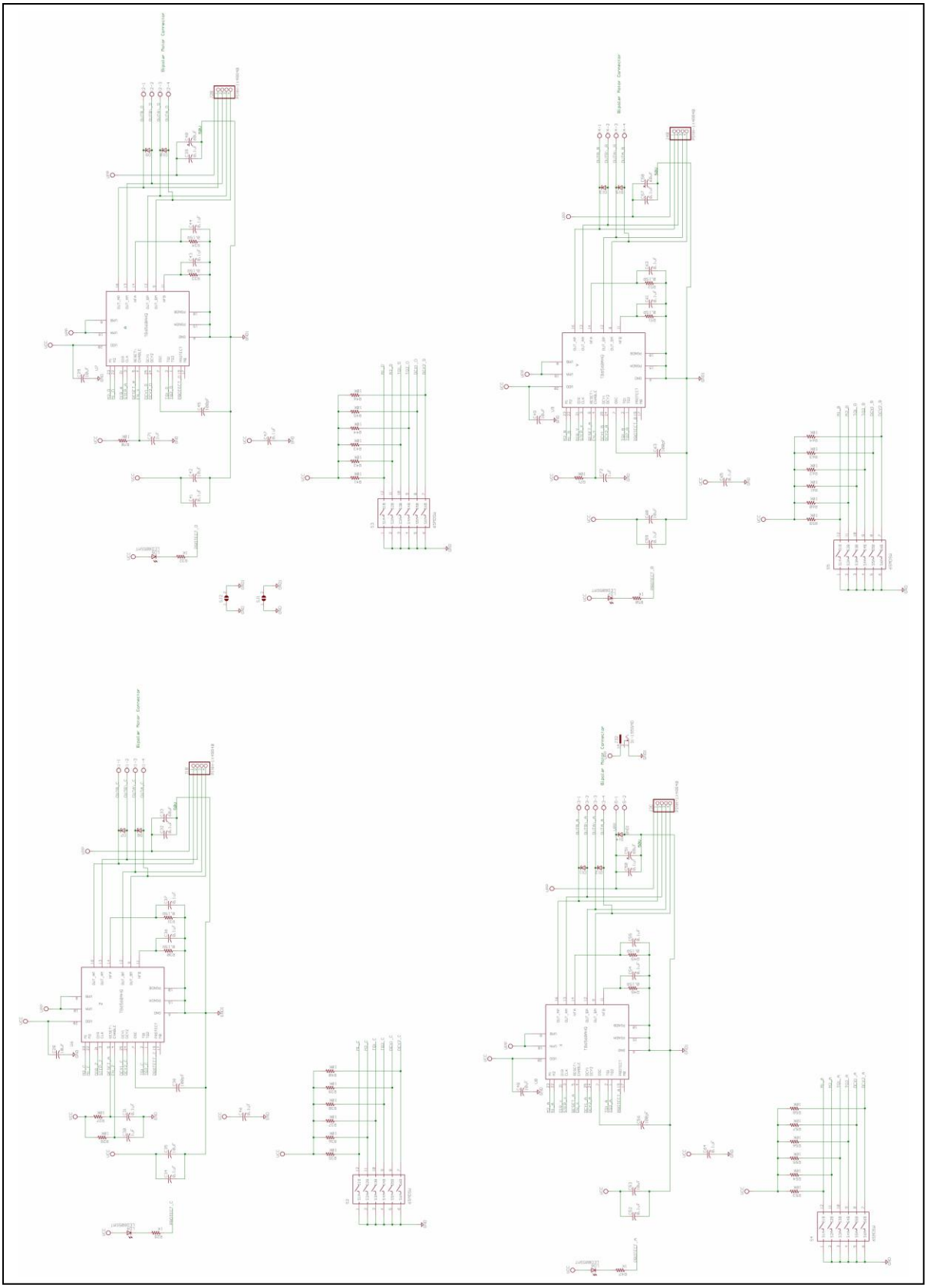
GenY32 Circuit Schematics

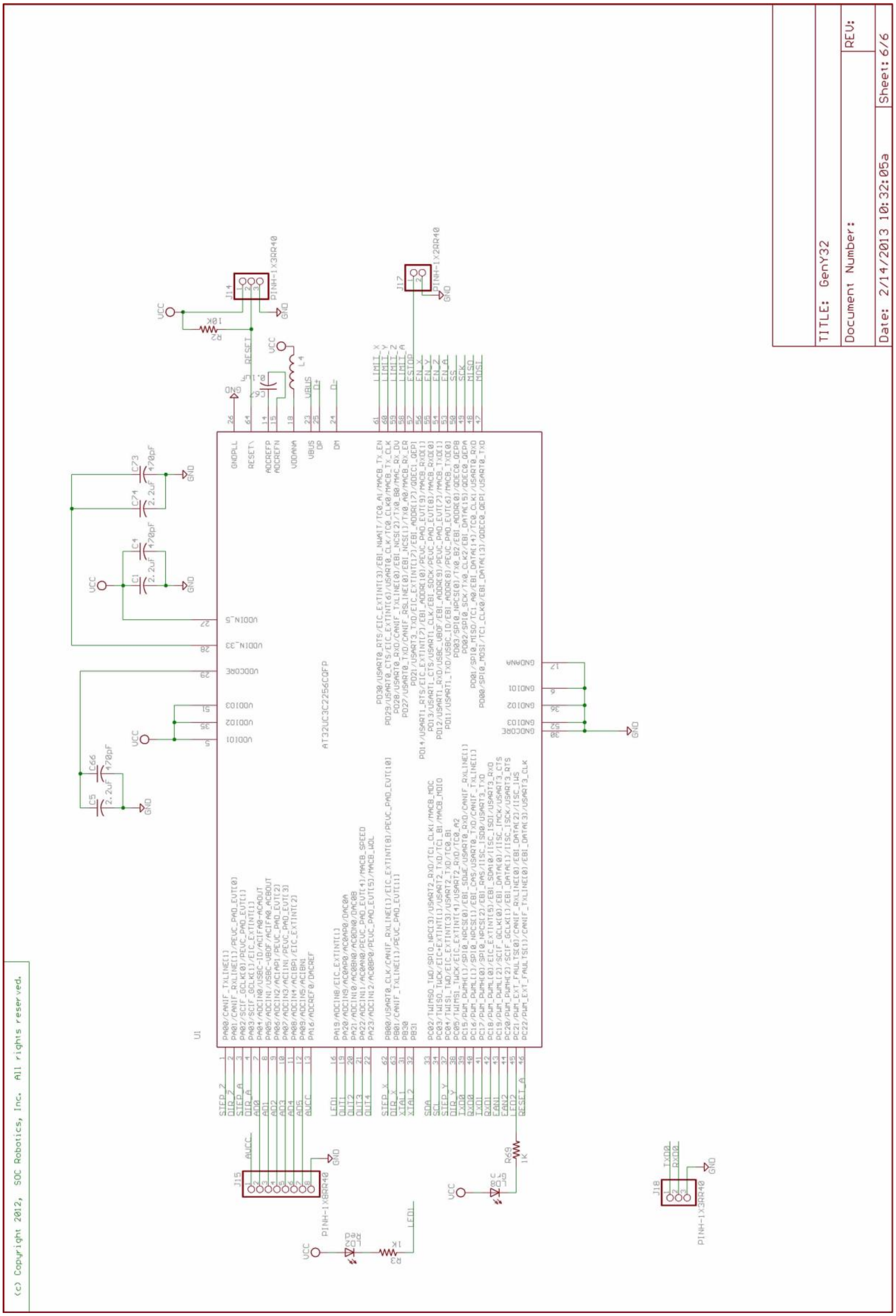



SOC Robotics, Inc
Vancouver, BC
TITLE: GenY32
Document Number: 20120417
REV: A
Date: 2/14/2013 10:32:05a
Sheet: 2/6
(c) Copyright 2012, SOC Robotics, Inc. All rights reserved.
I2C Port
USB-MINI
MICROSD


SOC Robotics, Inc
Vancouver, BC
TITLE: GenY32
Document Number: 20120417
REV: A
Date: 2/14/2013 10:32:05a
Sheet: 3/6
Limit Switch Ports
AUX OUTPUT PORT
45v max
LIMIT_X
LIMIT_Y
LIMIT_Z
LIMIT_A
ESTOP
VCC
GND
VPP
(c) Copyright 2012, SOC Robotics, Inc. All rights reserved.


Fan Power Connector
J10
PINH-1X2R40
VCC
GND
C20
1uF
LED GREEN
D3
R26
1K
C27
2.2uF
C24
10uF
D4
C69
0.1uF
R23
11K
R25
3.6K
10uH
L3
STPS2L40U
D6
U5
VIN
VREF
SYNC
COMP
INH
OUT
FB
GND
L5970AD
R24
4.7K
C26
220pF
22nF
C25
C23
4.7uF
C22
4.7uF
C21
100uF
VDD
J17B
PINH-1X2R"
SOC Robotics, Inc
Vancouver, BC
TITLE: GenY32
Document Number: 20120417
REV: A
Date: 2/14/2013 10:32:05a
Sheet: 4/6
(c) Copyright 2012, SOC Robotics, Inc. All rights reserved.




(c) Copyright 2012, SOC Robotics, Inc. All rights reserved.
AT32UC3C2256CQFP
TITLE: GenY32
Document Number:
REV:
Date: 2/14/2013 10:32:05a
Sheet: 6/6

Notes: