System Overview:

The system has:

- 5 dedicated outputs
- 5 dedicated inputs
- 8 reconfigureable I/O
- 1 UART
- 1 I2C
- 2 SPI (each uses 1, input, 1 output, and 1 reconfigurable)
- 6 Analog Inputs
- 3 Encoders
- 6 Stepper outputs
- 12 limit inputs
- 3 PWM outputs

This system takes inputs from either on board (USB, I/O, communication protocols) or TCP to control the stepper motors and output pins. Configuration xml files on the system govern the setup and input mapping. These files can be edited over the TCP interface or through a web browser at <ip address>/ConfigEditor.html (ex. 192.168.1.100/ConfigEditor.html). Once configured the system does not need to be configured each time. A demo UI is available at <ip address>/GUI_Main.html.

Getting Started:

1. Disconnect the debug jumper.
2. Connect the beaglebone ethernet to your computer's network
3. Then apply power to the beaglebone, note you will need more than just the usb power if you are connecting the spacemouse directly to the beaglebone. Use either a 1A barrel jack connector, or run regulated 5V 1A into the beaglebone header pins (see pinout below).
4. Once booted up you should see the beagle bone LEDs flash and do a scroll pattern
    a. Then they will switch to blinking the IP address
    b. Each LED is a bit in binary, LED 0/D2 is the 1's, upto LED 3/D5 is the 8's
    c. They will display each digit for 1 seconds
    d. All four lights on is a period
    e. A zero is all off for one second
    f. Once it finishes it will flash and scroll again, then repeat the IP address
5. Once you have the IP Address open a web browser and go to: <ip address>/ConfigEditor.html
6. Configure the board (see configuration section below for more details)
7. Next power off the board
8. Connect the debug jumper

9. Power up the board
    a. This time the IP address will not keep blinking
    b. This time the internal demo gui will be launched
    c. This will allow you to test I/O, move motors ect.
10. Go to <ip address>/GUI_Main.html
11. You can test out the configuration here
    a. If you need to make any changes go back to <ip address>/ConfigEditor.html
    b. Save any changes
    c. Then from the GUI_Main screen click on System Reset, this will apply any changes to the system, wait for the system to reconnect
12. Once you have your own remote TCP system connected and controlling it you can either leave this demo/debug gui running by leaving the jumper in on power up, or disable it by disconnecting it

System Function:

On startup it waits for the specified number of connections (in the System config) to connect then it will start. The system operates in different modes, in which each input can be mapped to different outputs or motors. Each mode needs to be explicitly requested to function. It starts up in a null state and must be sent a command of which mode to change to (future plan is to have an auto start mode defined). Then it maps each input to the outputs. This mapping is defined in the System Behavior tab. For inputs that are being sent into the system over TCP they require a timeout value to be included, if another input command has not been received within that timeout the system goes into an estop mode. In the case of a jog command it simply stops the motion.

It also sends out a ping to each remote connection, if the ping drops it will throw an error. This is automatically handled by SpacmouseCom's parseRecv() function, it just needs to be called at least 5 times pers second.

The SpacemouseCom.h file contains details on each function call. The receiving and parsing of data is handled in SpacmouseCom's parseRecv(). It will return a value for a "callback" like behavior based on what command was received. Each call will read in at most 1 command in.

Inputs for motor movement can be routed through the TCP connection at a rate of about 50Hz, internally a Spacemouse can be used at a rate of 200Hz. Each input can be rescaled via the on board configurable "Signal Conditioners" to allow for re-mapping or scaling input values before they are used.

The system can be pulled to see input/output values, motor state, and such.

Configuration:

You must press update for the changes to be available in the interface, and save to store the changes on the system. Changes will be allied on the next reboot or the next system reset command

System Behavior/Architecture:

This tab defines the mapping of inputs to outputs and motors. Note that the other tabs define what options are available for this mapping.

User Mode: is the name used to call up that mode when in operation

PRU Mode: is the mode that the internal hardware is operation in

- None: nothing can move or happen
- Jog: Inputs to axis are treated as speed values
- Init/Home: for changing configurations and homing
- Position Follow - Rel: treats inputs to the axis as position targets, will treat the first value in as the current position, then each one after that is a movement from that point
- Position Follow - Abs: treats inputs to the axis as position targets, will treat the first value in as the first target position and try to move to there
- Trajectory: work in process, not functional, the plan is for this to allow coordinated motion along either a waypoint path, or to a destination

Triggered By: Work in process

Mappings:

Mappings are the connection of inputs to the outputs and any signal conditioners that will be used in between.

Input:

Input device, ex the translation x axis of the spacemouse, or a digital input, ect.

Raw Variable:

Variable name of the input value

Sig Cond:

Signal conditioner to remap the input value, can be none

Mapped Variable:

Variable name after getting remapped

Output:

Output device, example Axis 0, or output 2, ect.

Once you change a mapping you must press Update Mappings to apply those changes, and then Save Mappings to store the updated config files on the system.

IO Devices:

These are the input devices. The External Input devices are remotely connected devices that will be providing inputs to the system. The Internal Spacemouse is for a spacemouse connected directly to the system usb port

External Input Devices:

These are the remote devices sending inputs over TCP.

Name:

The unique max 3 character name of this input device

UID:

Unique identification number for this device

Continuous (Bounded) Inputs:

Number of discrete inputs that are always going to be within a range.
Will appear in the input list as Name_C#

Continuous (Unbounded) Inputs:

Number of discrete inputs that are not constrained within a range.
Will appear in the input list as Name_U#

Binary Inputs:

Number of binary (values of 1 or 0 only) inputs
Will appear in the input list as Name_B#

Enable:

If checked the input device is used and its inputs are available.

Internal Spacemouse:

Name:

The unique max 3 character name of this input device

Axis ** or Btn *:

If this axis/btn is going to be used or not
Will show up as: Name_T*, Name_R*, or Name_B*

Enable:

If checked the input device is used and its inputs are available.

GPIO:

GPIO are the on board I/O. Note that GPIO_6-17 are the axis limits, GPIO_0-5 are input only, GPIO_26-31 are output only, and GPIO_18-25 are input or output.

Name:

Unique name of the input/output

Mode:

If the GPIO is used, and in what direction

PWM is a work in process

Motors:

This is for configuring the motor settings.

Name:

Unique name for this axis.

Axis:

Which Axis this is (0-5).

Speed Min:

Speed at which below this value it is set to 0 (units are mm/sec).

Speed Max:

Max speed, any values higher than this will be capped (mm/sec).

Software EOT Positive:

Max movement in the positive direction after homing that the axis is allowed to move, can be enabled/disbaled in the error mask.

Software EOT Negative:

Max movement in the negative direction after homing that the axis is allowed to move, can be enabled/disbaled in the error mask.

Reverse Direction:

Reverse all movement.

Reverse Homing:

Reverse the homing direction.

Reverse Fine Home:

Reverse the direction of motion for the portion of homing related to a fine home.

Home Speed:

Speed to home at (mm/sec)

Home Type:

What type of home movement.

Currently only support Flag Only (move to limit, backup slowly until untripped).

Steps per Rev:

Number of steps per revolution.

Steps for MM:

Steps per mm of motion.

Error Mask:

Bit mask of what limits the range of motion

- Bit 0: Pos limit sensor enabled
- Bit 1: Ned limit sensor enabled
- Bit 2: Pos software EOT limit enabled
- Bit 3: Neg software EOT limit enabled
- Bit 4: Coarse position check enabled
    - Coarse position check uses a signal that once per revolution trips and compares the expected step count to where it tripped to verify that it has not lost counts, not meant to be highly accurate, need a large trip zone.

Limit x Pin:

Pin number that is used for limit in that direction

Limit x Act:

If check signal is treated as active high, if unchecked it is active low

Home Pin:

Pin to home to

Home Act:

If home pin is active high or low

Fine Pin:

Pin for a fine flag

Fine Act:

If Pin is active high or low

Pos Follow Max Dist:

Leave at 500000 for now, work in process

Signal Conditioners:

These can be used to remap input values. One signal conditioner can be used to scale multiple inputs.

Bounded Scale:

Name:

Unique name

In Min:

Minimum value of input, anything lower than this is set to this

In Max:

Maximum value of input, anything larger than this is set to this

Out Min:

Lowest value that the output is mapped to

Out Max:

Largest value the output is mapped to

Exp:

Exponent of the mapping function, 1 is linear

Unbounded Scale:

Name:

Unique name

Scale factor:

Value to scale the input by

Continuous to Binary:

For mapping a continuous value to a binary (0 or 1) value

The set and reset conditions will change the state of the output when the condition is met.

System:

Number of connection:

Number of devices expected to be connected to this system.

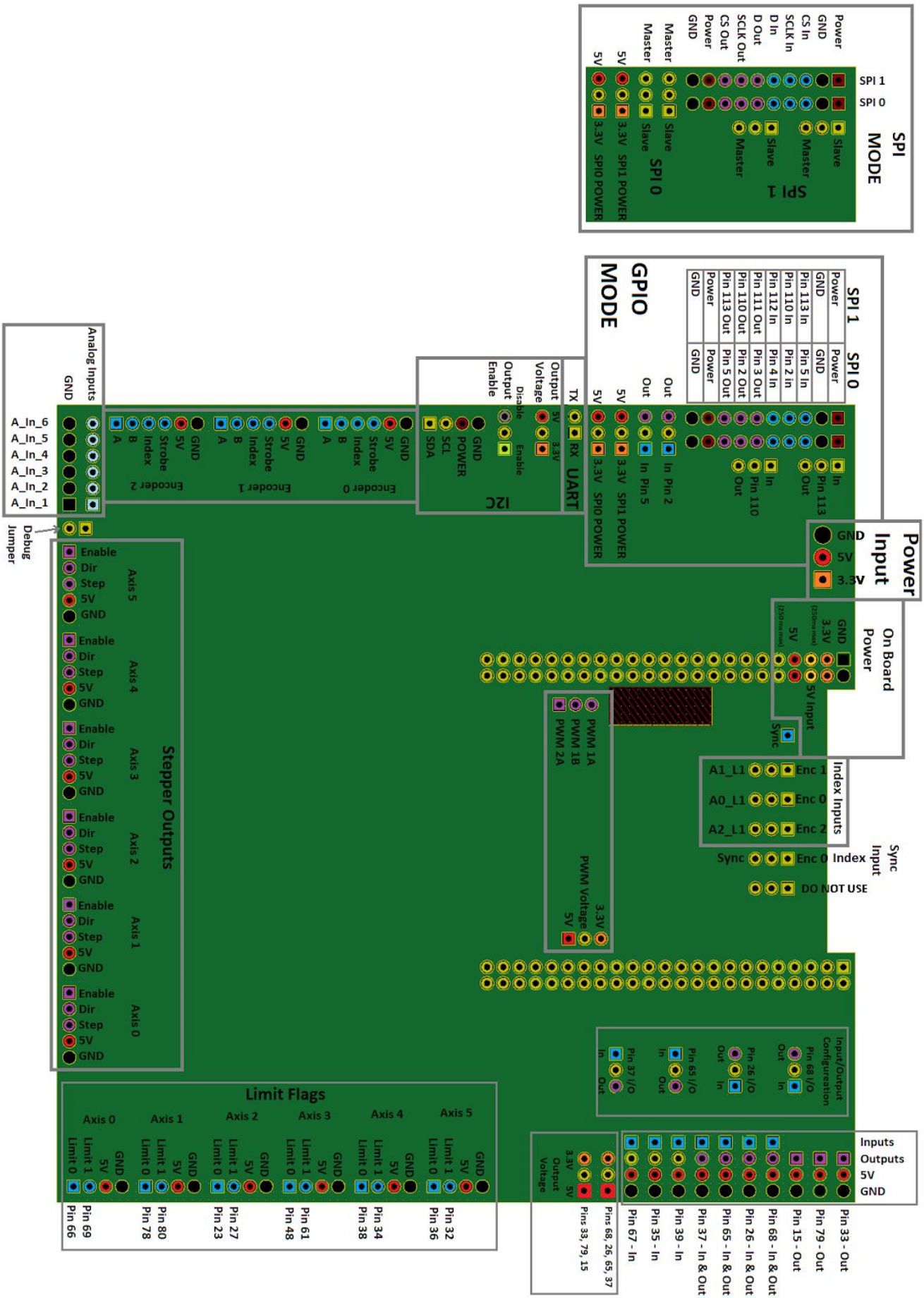
DHCP:

If checked IP address is assigned via DHCP, otherwise it uses the static IP set below

IP Address:

Static IP Address to be used.

Pin Mapping:


SPI MODE
SPI 1
SPI 0
Power
GND
CS In
SCLK In
D In
D Out
SCLK Out
CS Out
Power
GND
Master
Slave
3.3V
SPI1 POWER
SPI0 POWER
GPIO MODE
SPI 1
SPI 0
Pin 113
Pin 110 In
Pin 112 In
Pin 111 Out
Pin 110 Out
Pin 113 Out
Pin 5
Pin 2
Pin 4
Pin 3
Pin 2 Out
Pin 5 Out
Pin 110
UART
I2C
TX
RX
Output Voltage
Output Enable
Disable
Enable
SCL
SDA
POWER
Encoder 0
Encoder 1
Encoder 2
Index
Strobe
Analog Inputs
A_In_6
A_In_5
A_In_4
A_In_3
A_In_2
A_In_1
Debug Jumper
Power Input
On Board Power
5V Input
Sync
PWM 1A
PWM 1B
PWM 2A
PWM Voltage
Index Inputs
A1_L1
A0_L1
A2_L1
Enc 1
Enc 0
Enc 2
Sync Input
DO NOT USE
Input/Output Configuration
Pin 68 I/O
Pin 26 I/O
Pin 65 I/O
Pin 37 I/O
Stepper Outputs
Axis 5
Axis 4
Axis 3
Axis 2
Axis 1
Axis 0
Enable
Dir
Step
Limit Flags
Limit 1
Limit 0
Pin 32
Pin 36
Pin 34
Pin 38
Pin 61
Pin 48
Pin 27
Pin 23
Pin 80
Pin 78
Pin 69
Pin 66
Inputs
Outputs
Pin 33 - Out
Pin 79 - Out
Pin 15 - Out
Pin 68 - In & Out
Pin 26 - In & Out
Pin 65 - In & Out
Pin 37 - In & Out
Pin 39 - In
Pin 35 - In
Pin 67 - In
Pins 68, 26, 65, 37
Pins 33, 79, 15