# STX Design and Programming Manual

**Spectrum Techniques Inc**

17

## Contents

## STX Single Channel Analyzer (SCA) design philosophy

The STX (ST365/375) is primarily a single channel analyzer (SCA), as such it counts all pulses coming from a radiation detector such as the Geiger Muller tube or a scintillation device.

The Geiger Muller (G-M) tube is a gaseous ionization detector and uses the Townsend avalanche phenomenon to produce an easily detectable electronic pulse from as little as a single ionizing event due to a radiation particle. It is used for the detection of gamma radiation, X-rays, and alpha and beta particles. It can also be adapted to detect neutrons. The G-M device provides no discrimination between radiation types; such as between alpha and beta particles.

A scintillation detector or scintillation counter is obtained when a scintillator is coupled to an electronic light sensor such as a photomultiplier tube (PMT), photodiode, or silicon photomultiplier. PMTs absorb the light emitted by the scintillator and re-emit it in the form of electrons via the photoelectric effect. The subsequent multiplication of those electrons (sometimes called photo-electrons) results in an electrical pulse which can then be analyzed and yield meaningful information about the particle that originally struck the scintillator.

The G-M tube and scintillation devices require a high voltage to operate that is provided by the STX. The devices provide pulses to the STX that correspond to a ionizing radiation event. The pulses are qualified at the STX SCA and counted.

There are three configurations of the present STX design:

1. ST365: Includes a Communications Board for talking to a PC. The ST365 often includes a Display Board for local/manual control the STX.
2. ST475: Has no communications for talking to the PC but is controlled by operator inputs through the Display Board.
3. **STXXX no Display Board and no Communications Board. An additional USB port is located on the SCA for communication and power.**

## Design Basics:

The STX is broken up into four subsystems for modularity, performance and EMI.

The four systems are as follows;

1. The SCA controller a single board that performs all of the actual SCA counting functions
2. The High Voltage (HV) power supply, which handles control and monitoring of the high voltage used to bias the detector.
3. The Communications Board, contains the Ethernet controller, WIFI controller and USB
4. The final is optional user interface which consists of keys and a display to allow a manual control (non PC) of the STX.

### Event driven system

In the STX system there are essentially three asynchronous events that drive the system.

1. Messages received on the the Ethernet, WIFI or USB communications Interfaces on the Communications Board (from the PC).
2. Keys pressed on the Display Board by an operator.
3. Pulses from the ionizing radiation detector.

These events are considered asynchronous because the main firmware has no control over when they will occur in time. The following is a brief discussion of how the STX design handles each:

### Communications

Communications protocol is discussed in detail in the protocol section, here we will just cover the basics.

## RS485 Power Distribution

Power distribution is integrated into the RS485 distributed network. All STX modules connect to the RS484 distributed network.

| Pin | Description | Wire Color |
|---|---|---|
| 1 | +5V dc power distribution | Red |
| 2 | +RS485 (signal) | Blue |
| 3 | -RS485 (signal) | Yellow |
| 4 | Common ground | Black |

***Table (1) PCB comm connector***

The 5VDC power is provided by the ON/OFF button which is located on the Display Board. The Display Board has one RS485 connector that provides power to the SCA Board. **The SCA Board acts as a hub with three RS485 connectors.**

**The USB on the SCA Board is an additional communications interface as it connects to the RS485 port. In addition. the SCA USB can be used to provide power to the STX modules (+5V @ 500 mA). This is used whenever a system is built that does not have a Display Board, otherwise it is not used.**

## Ethernet and WiFi

The Tiva (TM) Ethernet controller provides a telnet service and opens an Ethernet to serial bridge on TCP/IP port 23. The Tiva (TM) controller on the Ethernet port bridges the UDP packets to the RS485 distributed network in the STX.

WiFi listens for UDP packets incoming on port 2020. When messages are fed to the WiFi CC3200 controller, it will bridge these messages to the RS485 distributed network.

A PC connected to the either the USB, WiFi or Ethernet ports acts as a bus master sending commands to the STX boards over the RS485 distributed network. The operator must select only one method of communication with the STX as multiple connections would cause communications collisions on the RS485 distributed network.

All STX boards listen for the communications messages and only respond if appropriate. No communications errors are reported as this causes unnecessary collisions.

Finally the Display Board acts as a local master that also communicates with the STX boards through the RS485 distributed network.

The SCA responds to data requests or commands equally. All packets are perceived with equal priority. Care must be taken not to confuse or overwhelm the STX by attempting to send packets from more than one source simultaneously. Don't connect using WiFi, USB and Ethernet at the same time. The messages are not buffered in a message queue. The final message received correctly by any board will be the only one that is executed.

**WARNING:** Don't connect using WiFi, USB and Ethernet at the same time.

Communications are interrupt driven and take priority of the main routine. The Interrupt Service Routine (ISR) saves the incoming message for processing by the main routine. If the packet is a command, the command will issue a state change in the state machine and the command will be executed. If the message is a data transfer, then the data will be parsed, translated from HEX ASCII string to a binary value and stored in the shared global database **(on the SCA uC???)**.

### Key press

The Display Board contains a keypad and display which are monitored by a Display Board microcontroller. Pressing a key will allow the user to locally select the function (HV, counts, time) displayed. In addition, the Display Board allows the operator to change an system parameters to be sent to either the HV microcontroller or the SCA microcontroller. Finally, the Display Board listens for data messages and commands from the other boards or external devices such as the PC. Typically the display on the Display Board mimics the display on the PC. However, the PC will not change the focus of the display on the Display Board. HV data sent will be stored internally, but will only be visible if the operator presses the HV menu key to select the HV value as the active displayed parameter.

### Pulses from the radiation detector

This is a priority event as the accurate counting and classification of signals from a detector is paramount to the STX. The SCA Board has a direct interface to the detector. The SCA uses two asynchronous counters that operate independent to the SCA main microcontroller. The counters are implemented on a second microcontroller on the SCA Board. This allows the pulses to increment the counters without interrupts or action from the main microcontroller. When the SCA is in counting state, it will periodically read both counter registers, immediately clear them to prevent over run, and add the captured value to the integral count in the system database. When a data read command is received from the master, the SCA response message will only include that latest value in the database, not the actual counter register. This prevents the request from interfering with the ongoing count.

The count value **(in the database??)** is a 32 bit word so it will hold more than 4.2 billion counts before overflow. **How many bits in each counter?**

Counter1 contains all counts higher than the low (noise) threshold

Counter2 contains all counts higher than the high (window) threshold

Differential counting (within the window) is left to the applications **(on the PC??**) by subtracting Counter2 from Counter1.

## STX Radiation Detector External Signal Connections

There are essentially two external connections:

1. the HV (high voltage) which consists of an MHV connector and has voltages present from 100VDC to a maximum of 1200VDC.
2. The BNC signal connection with the pulses from the radiation detector.

   There are two methods to connect radiation detectors to the STX:
   **1. One Wire Connectivity (G-M Tube) and,**
   **2. Two Wire Connectivity (scintillation device).**

### One Wire Connectivity

In a one wire mode, where the HV and pulse signals share the same path, only the MHV connector is used. If a Geiger Mueller tube with a BNC is used in one wire mode , then an MHV to BNC adapter cable is required, **HV will never appear on the BNC because it is only safety rated for 500V max.????????**

In one wire mode the signal and HV share the connection. Internally the signal is separated from the HV through an internal blocking capacitor and the signal is switched to a connector bringing it over to the SCA Board. If external signal is required it will be now present on the BNC. Otherwise there is no connection needed for the BNC.


**Figure X. One Wire Connectivity – G-M tube: HV and pulses on MHV**

## Two Wire Connectivity

In Two Wire mode, two cables are used (Scintillator configuration) The MHV has HV 100VDC to 1200VDC and the signal (5V pulses) is on the BNC only, a G-M tube will need a MHV to BNC adapter cable.

**Figure X Two wire Connectivity - Scintillation: Pulses on BNC only.**

**WARNING:** connecting the Scintillation device in One wire mode will damage the STX requiring the unit to be returned for repair. **Not sure if this voids any warranty????**

# STX Comm Interface and Command Protocol

## Overview:

The STX contains four distinct subsystems:

1. The SCA (Single Channel Analyzer) board: pulse counter.
2. An optional Display Board: LCD display. keys and knobs user interface.
3. An optional Communications Board: USB, 10/100M Ethernet, and WiFi.
4. An HV (High Voltage ) Board: 100 to 1200V DC @ up to 6mA

The ST475 **(475??)** will not contain the Communications Board. The ST365 will contains all four boards. **A third option includes only the SCA and the HV Boards.**

Communications between the boards is facilitated through a RS485 distributed network using the following parameters:

1. Baud Rate 115200 bps - allows over 120 data exchanges per second.
2. 8 bits
3. No parity
4. Single stop bit

Typically the PC is the bus master. However, the Display Board serves as a master when the operator is pressing keys. This means that commands will originate from the Display Board.

The SCA Board and the HV Board are slave devices. This means they will not initiate a command or data request, but only respond to command placed on the bus by either Display Board or the PC.

## USB Interface

USB is dedicated as a serial port. There is no controller but a single FTDI chip that performs the conversion from USB to the RS485. This allows the USB to have a direct connection to the STX RS485 distributed network.

## Ethernet Interface

10/100M Ethernet has two TCP/IP services:

1. HTTP (internally generated web page for network connection parameters and serial port configuration). The configuration parameters should not be changed to insure operation. Static or DHCP operation is possible. The configuration page is available using the assigned IP address for the Ethernet port . This can be obtained by checking the device on the attached device list on the network switch or router. The STX device will be listed as ST365 Telnet Server. Entering the ST365 IP address into the browser (Explorer or Chrome) will bring up the ST365 configuration web page.
2. A Telnet Server will respond to connections on the assigned TCP/IP port and open a telnet socket between the PC and the ST365. The socket will be processed by the Tiva (TM) microcontroller and provide direct access to the STX RS485 distributed network.

## WiFi Interface

The 2.4 GHz WiFi is an access point (AP) capable of handling up to five WiFi devices. In WiFi mode, the ST365 identifies on the WiFi available devices list or scan list as ST365_001. The last 3 to 5 numerical digits (001) representing the serial number. **Alternatively, it can be initialized as a connected device on the WiFi Network attached to a common router or access point??????** The ST365 uses a Texas Instruments CC3200 WiFi microcontroller, so the TI simple link provisioning tools can be used to develop an smart phone application that will find the ST365 device and allow the user to associate the ST365 to a network or router.

Once an ST365 is attached to a WiFi network, there are two interfaces available;

1. An HTTP web server that allows the user to configure the network parameters and status, and simple control , start / stop and report count data
2. A UDP server that broadcast UDP packets. An application for a tablet or PC can be written that listens on the IP address and assigned UDP port, and displays the received data. In addition, the ST365 listens on a UDP port for packets that commands and passes the commands to the STX RS485 distributed network. **The UDP port is available on the device set up as a connected device (non AP, also known as Direct Connect) only, and can accommodate as many**

**connected devices as can listen.** UDP is a broadcast protocol with no guaranteed delivery.

## Communications Protocol

The communications protocol is a hexadecimal ASCII based data exchange with fixed sizes, Commands used to initiate action or request data transfers are limited to 4 characters. Data transfers can be of three distinct types:

1. Parameter exchange, systems settings and operational parameters,
2. Data exchange, counts , rates and sample feedback values,
3. System exchange, model, serial number, MAC address, boot count and system specifics.

## Command and Response Protocol

All commands are 4 character ASCII strings. Of the form “>XX\r”. The “>” character is the start character .

“XX” is hexadecimal ASCII of the range 0x00 to 0xFF, representing a decimal value from 0 – 255 identifying the command issued.

“\r” is a simple ASCII carriage return (Hex value (0x0d) or decimal (13) indicating end of the string.

Response strings all start with “#”.

STX boards listen for ‘#’ response strings to update any changes to system operating parameters. If the attached modules do not need the transmitted response string data, it is simply ignored.

**NOTE:** Commands should not be stacked or sent without waiting for the appropriate response. Otherwise the system will experience communication collisions.

### SCA Command List

```
• COMM_CMD_RST    0x00      // Reset all
• COMM_CMD_STRT   0x01      // Start counter
• COMM_CMD_STOP   0x02      // Stop counter
• COMM_CMD_STAT   0x03      // Request status
• COMM_CMD_CNTS   0x04      // Request counts
• COMM_CMD_PARM   0x05      // Request parameters
• COMM_CMD_SYS    0x06      // Request system parameters
• COMM_CMD_STOR   0x07      // Store current parameters to eeprom
• COMM_CMD_DEMO   0x08      // Start Demo Counter
```

***">01\r"*** **Start Counter** – resets all internal values ramps up the high voltage and starts the counter. There will be a delay to ramp up the high voltage, counting will begin with the transmission of a count data packet, until a stop counter is received or internally generated. The ramp time for 0 to 1200V is approx 5 sec.

***">02\r"*** **Stop Counter**- This stops the current operation and transmits the last captured count data packet **(is this the final accumulated count from the database?)** after stopping the sequence. Sending a second stop command will reset the count values to zero.

***">07\r"*** **System and control parameter store.** This command forces a transfer of the current operational settings and parameters to the internal non-volatile memory backup. This is used to permanently store system data such as serial and model numbers as well as temporary storage of current settings allowing the STX to reboot to it last know values. In practical use this is not used as the new settings will become factory defaults**????**.

**NOTE:** The operator (via the ST365 Display or PC) should program the parameters settings on start up.

***">08\r"*** **Start Demo Mode** same as Start Counter but connects the counters to the internal 32KHz clock. This will simulate a count without a source detector for demonstration and repeatable result testing.

## SCA Data Transfer Requests

***">03\r"*** **Request Status** (counter activity) used to verify status of STX. Return values will be six characters in the form "#03XX\r where "#" is the ACK start char "03" is the read back command issued and "xx" represents the status codes . The sixth character is the "/r" carriage return

***">04\r"*** **Request the Count Data** generates a read and transmission of the latest count data.

***">05\r"*** **Request the Parameters** Request operational parameters , this is typically used to get the current parameters from the counter. If an externally connected device or the Display Board has new parameters a packet will be sent without issuing this command. This indicates another device is sending the system new parameters. The counter will receive this and validate and set the new parameters.

***">06\r"*** **Request the System Data** generates a read and transmission of system specific data , this should only be used if the STX is in Idle waiting mode or after program start up to identify the connected device.

## SCA reply to a ">06/r " command

Only the SCA Board will respond to this command as it will hold the system data. The system data response is Hex ASCII string 28 chars long with fields defined as follows:

"#06BBBBBBBBMMMMSSSSCCCCEEEE\r"

- **06** is the returned command "06"
- BBBBBBBB is the boot counter , a Hex ASCII long integer (32 bit) representing 0x00000000 to 0xFFFFFFFF, incremented whenever the STX is powered up.
- MMMM is the ST365 system model number, a Hex ASCII integer (16 bits) representing 0x0000 to 0xFFFF a unique number representing how the STX is configured and sold. Set at time of manufacture only.
- SSSS is the system serial number, a unique Hex ASCII integer (16 bits) representing 0x0000 to 0xFFFF set at the time of manufacture only.
- CCCC is the system configuration, the type of system (display, ENET, etc) a unique Hex ASCII integer (16 bits) representing 0x0000 to 0xFFFF, set at the time of manufacture to identify module configuration.

- EEEE is the EEPROM byte size , a Hex ASCII integer (16 bit) representing 0x0000 to 0xFFFF set at the time of manufacture to indicate available non volatile memory for internal use.

### SCA reply to “>04/r” command

Only the SCA Board will respond to this command. The reply is a hexadecimal ASCII string 36 characters long with four 32 bit hexadecimal ASCII fields defined as follows:

“#041111111122222222RRRRRRRREEEEEEEE/r”

- 11111111 in the response string is the **lower threshold** total count value in raw counts . This is a Hex ASCII long integer (32 bit) representing 0x00000000 to 0xFFFFFFFF full scale.
- 22222222 in the response string is the **upper threshold** total count value in raw counts , a Hex ASCII long integer (32 bit) representing 0x00000000 to 0xFFFFFFFF full scale. Count 2 must be subtracted from count 2 to get the SCA windowed counts.
- RRRRRRRR is the rate count value in counts per second , a Hex ASCII long integer (32 bit) representing 0x00000000 to 0xFFFFFFFF
- EEEEEEEE is the elapsed timer value in increments of 25ms, representing the internal ST365 state machine timing. Hexadecimal ASCII long integer (32 bit) representing 0x00000000 to 0xFFFFFFFF

### *SCA Reply to a “>05/r” command*

Sending this packet out before a start count command sets the new parameters to be used by the counter. It consists of 25 chars. **So the SCA generates a unsolicited response string???**

“#05RRRR11112222FFFFCCGG<cr>”

- 05 is the returned request command

- RRRR is the sample reset timer in seconds , a Hex ASCII integer (16 bit) representing 0x0000 to 0xFFFF the counter board will automatically stop and issue a stop command when this value is reached by the elapsed timer during counting.
- 1111 is the lower comparator threshold value 0 – 4500 mV , a Hex ASCII integer (16 bit) representing 0x0000 to 0x1194 any value above 4500 (decimal) will result in the factory default being used instead by reading the EEPROM value.
- 2222 is the upper comparator threshold value 0 – 4500 mV , a Hex ASCII integer (16 bit) representing 0x0000 to 0x1194 any value above 4500 will result in the factory default being used instead by reading the EEPROM value.
- FFFF is the fine gain value value 0.5 – 1.5 x’s , a Hexadecimal ASCII integer (16 bit) representing 0x01F4 to 0x05DC , **in non SCA function devices such as the STX??????**, this is nominally set to 1000 or a gain of 1x’s
- CC is the input channel number 0x00 for GM Tube , and 0x01 for Scintillator.

  **NOTE:** If the value is above 0x01 the internal factory default is used.

- GG is the input channel Gain value between 0x00 and 0x07 , a Hex ASCII char (8 bit) representing 0x00 to 0x07 , if the value is above 0x07 the internal factory default is used, in a STX this is typically 0x00 for a gain of 1. In SCA mode the following values are mapped as follows:

| | |
|---|---|
| 0x00 | - gain of 1x’s |
| 0x01 | - gain of 2x’s |
| 0x02 | - gain of 4x’s |
| 0x03 | - gain of 5x’s |
| 0x04 | - gain of 8x’s |
| 0x05 | - gain of 10x’s |
| 0x06 | - gain of 16x’s |
| 0x07 | - gain of 32x’s |

## HV Command List

- `COMM_CMD_HVOK 0x10 // HV is ready` **(appears more like a status reply??)**
- `COMM_CMD_HVRM 0x11 // HV is ramping trigger wait` **(appears more like a status reply??)**
- `COMM_CMD_HVON 0x12 // HV On`
- `COMM_CMD_HVOF 0x13 // HV Off`
- `COMM_CMD_1WON 0x14 // One-wire mode HV switch on`
- `COMM_CMD_1WOF 0x15 // One-wire mode HV Switch off`
- `COMM_CMD_HVST 0x16 // HV Status Request`
- `COMM_CMD_HVRD 0x17 // Read HV data`

These commands tell the associated systems that a desired condition has been reached without transmitting the entire status string.

***“>10\r”*** **HV Ready** sent by the HV controller to indicate that the High voltage is at the desired set value and ready . This signal command should only be sent by the HV controller, and not user ap-plication software as that would indicate a false HV ready signal.

***“>11\r”*** **HV Ramping** sent by the HV controller to indicate that the High voltage is ramping up to the desired HV set point and is not yet ready to use.

### *HV Commands:*

***“>12\r”*** **HV On Command** This is a user issued command that turns on the HV . This command starts the initial ramp up of the HV if completely off. Usually needs to be called after boot and only if the current state is wait for ramp mode 0x01 when HV state read shows that HV ready and Idle .

***“>13<CR>”*** **HV Off Command** Turns the HV off.

***“>14\r”*** **HV One-Wire On Command** Turns on the HV switch connecting high voltage to the detector BNC connector, this should only be used in a one wire mode, if used during a two wire (separate signal and HV) connection, the detector could be damaged.

***“>15\r”*** **HV One-Wire Off Command** Turns the HV switch off setting a separate HV connection from the signal connection.

## Command Reply Protocol

All replies start with the acknowledge start char “**#**”, and are identified by the first two ASCII characters representing the originating command. The receiving software should read these two ASCII characters to identify the data in the response.

**HV Status Request  Reply**

- “#1601\r”  HV is off and Idle waiting for start command
- “#1602\r”  HV is Ramping to target voltage
- “#1603\r”  HV is on and settled

**HV Data Request Reply**

- “#17vvvvffffppppss\r”  HV data the return value is the target voltage, current voltage, PWM setting and status

  “vvvv” **target volts**  represents 0x0000 – 0xFFFF  (50 – 1200 VDC)

  (0x0032 – 0x4B0) VDC

  “ffff”  **feedback volts**  volts  (50 – 1200 VDC)

  “pppp” PWM value representing the pulse with or duty cycle.

  “ss”  HV status flags  The upper two chars are always zero

| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|
| unused | unused | unused | HV FLT | 1 Wire on | HV OK | HV RMP | HV ENA |

- 0x01  HV is enabled and idle (HV is currently off), the enable flag shows that the HV controller is booted and waiting.
- 0x03  This HV is enabled and ramping (takes between 2 and 4 seconds)
- 0x05  This HV is enabled and at target value ready to use

Bit 3 is the status of One-wire or Two-wire switch , a “1” bit in this position indicates that the One-wire mode is enabled and HV is switched onto both the HV connector and the Amplifier input. Default is One-wire Off to protect the sensors **(what sensors??don’t you mean to protect the radiation detectors????).**

This bit is switched on and off only by issuing a 1 wire on command “>14\r” command to turn on, and a 1 wire off command “>15\r” to turn the switch off, (two wire mode).

**HV Data Request “>17/r” Not sure why this is covered twice, possibly an earliar revision that was not deleted???**

Only the HV Board responds to this command as it contains the high voltage access and control. The response string is “#17VVVVSSSS/r” 12 characters

- VVVV is the high voltage value 0 – 1200V , a Hex ASCII integer (16 bit) representing 0x0000 to 0x04B0 any value above 1200 will result in the factory default being used instead by reading the EEPROM default value.

- SSSS is the high voltage Status flags used to control how the HV is used. The flags reside in an integer structure 0x0000 in binary xxxx xxxx xxxx xxxx **As described previously, this is a 8-bit register which shoud only need to hexadecimal characters???**

Bit 0 xxxx xxxx xxxx xxx1 represents the HV Enabled flag, this bit will indicates that the HV controller has turned on the HV.
“>0x12/r” (HV On) and “>13/r” (HV Off)

Bit 1 xxxx xxxx xxxx xx1x represents a quick HV ramp flag, if set this bit is set then the desired voltage is not yet reached and the system is ramping

xxxx xxxx xxxx xx1x (HV Ramping ) and xxxx xxxx xxxx xx0x (HV no longrt ramping)

Bit 2 xxxx xxxx xxxx x1xx represents a quick HV is OK flag, if set this bit is set then the desired voltage has been reach and the HV is settled.

xxxx xxxx xxxx x1xx (HV One Wire ON ) and xxxx xxxx xxxx x0xx (HV One Wire Off)

Bit 3 xxxx xxxx xxxx 0xxx represents the HV One wire on flag, this bit will indicates that the HV controller has turned on the HV one wire connection switch. A short cut command is given to issue this HV One Wire command without transmitting the HV status word every time.

“>0x14/r” (HV One Wire ON ) and “>15/r” (HV One Wire Off)

## Typical Program Flow

Only issue commands, transmission of parameters is done just before issuing a start count command.

The program should listen to the communications channel and update the display accordingly. Messages will only be available on the RS485 distributed network when a master (Display Board, or PC) initiates a command or request. Most responses will come from the SCA Board. If more than one reply is required, there will be an approximate 250ms delay between transmissions. The delay allows the microcontrollers to complete their task before receiving new commands. There is a message buffer on the microcontrollers, but no message queue. However, as the message are interrupt driven, the ISR will over right the currently buffered message if a new message arrives. If verification of receipt of a message or command is required, use the "**>**" commands to request a resend of the data, **it will only include values that have been stored in the local memory and not a repeat of the communications buffer????**. The status request command is the best approach for determining the status of a module.

The count data is sent out with corrected time values on an internal **300ms** update rate. This update rate is set by the time to process counts and transmit the data string before switching the states in the main routine's state machine.

Monitoring the distributed network will allow the connected device (PC) to update the screen in real time. If a response is not forthcoming wait one cycle (250ms) and try again.

**NOTE:** Avoid constant polling of data, if a response is not received back, it represents a communications error, try again after a slight delay.

The following is a series of example sequences to correctly describe the proper methods to achieve successful data transfer and control of the ST365. In all cases the use of the Status Request command ">03/r" is the proper method to determine how to proceed. The Status Request will return the status of the SCA state machine and if the SCA is ready to proceed.

**NOTE:** Reading the status of either the SCA or HV controller before polling is advised as this will indicate that the system is ready to perform the function required.

## Status Request Reply

The Status Request “>03/r” will return as “#03xx/r” where the xx represents the current status code.

- “#0300/r” status code 0x00 the SCA system is **booting** and initializing, chances are that the firmware has already completed this mode and has moved on to the ready idle state before this command can be sent.
- “#0301/r” status code 0x01 the SCA system is **ready and idle**, waiting to receive parameter settings and control commands (the HV is off).
- “#0302/r” status code 0x02 the SCA has received a **ramping** status from the HV controller.
- “#0303/r” status code 0x03 the SCA system is **ready** and waiting to start counting.
- “#0304/r” status code 0x04 the SCA system has received a start count and is **initializing the counters**, chances are that the firmware has already completed this mode and has moved on to the counting state before this command can be sent and processed.
- “#0305/r” status code 0x05 he SCA system is **stopping**, chances are that the firmware has already completed this mode and has moved back to either the waiting state 0x03 . if the HV is still enabled or the idle state (0x02) indicating HV has been turned off before this command can be sent and processed.
- “#0306/r” status code 0x06 the SCA system is **counting**.
- “#0307/r” status code 0x07 the SCA system is **storing the parameters** into EEPROM, this takes some time so the user must wait for a return to IDLE or Wait modes before proceeding to other commands or data requests.
- “#0308/r” status code 0x08 the SCA system is **counting in demo mode** no detector is used, counts come from internal 32Khz clock source.

# Programming Examples

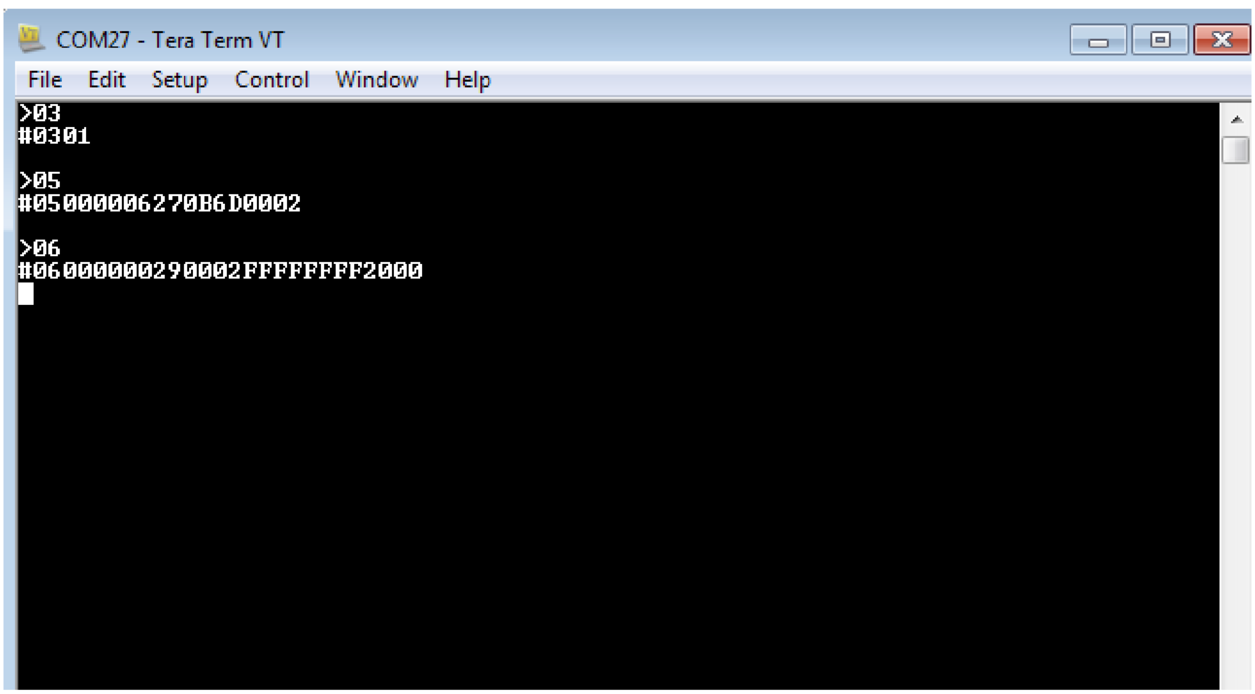
## Initialization

The SCA is ready to communicate after a re-boot when the "#0301/r" reply is received in response to a status request. Next, the stored parameters from the EEPROM on the STX need to be shared with the PC. So a read parameters command is issued ">05/r".

Step 1: send status request **">03/r"** and poll until idle ready state is reached.

Step 2: send a parameters read request, **">05/r"** command to retrieve the ST365 parameters.

Step 3: The user software may want to identify the model and configuration of the STX instrument connected, so an optional third read request command is typical in a PC connected device. The command "**>06/r"** is issued to retrieve the model, serial number and factory only set codes from the ST365.




**it would be nice to invert the colors in this image so the printer doesn't run out of black ink . And not to mention it would just look better.**

## Start a count cycle

Check and or initialize the HV, by polling the HV Status

**Step 1)** Issue an HV status request **“>16\r”** to check the status of the HV controller, the status string determines the next step.

Possible return values:

- “#1600/r” HV Board is booting
- “#1601/r” HV is Off and Idle waiting for HV On command
- “#1602/r” HV is currently ramping to target (2 – 4 seconds)
- “#1603/r” HV is on and active (ready to count)
- “#1604/r” HV has had a fault (reboot to clear) service if this persists

A return of “#1603/r indicates the system is ready to count .

If the HV is off “#1601/r” perform the following sequence

- Issue an HV On command **“>12/r”**
- Issue Read State **“>16/r”** wait for a response Read HV status to wait for ramping complete and HV Ready response of “#1603/r”

**Step 2) Verify the HV setting**

- First method, simple verification : read the current HV target voltage by issuing a HV read data command **“>17/r”** , the fist ASCII integer in the response is the target voltage setting. In 1 V steps from 50 – 1200 Example: “#17VVVVFFFFPPPPSS\r” VVVV is the hex ASCII integer value 0x0032 – 0x04B0. Zeros can be sent for the FFFF PPPP and SS as these values are not written built generated locally in the HV controller.

- Second Method (preferred ) Send a new Target voltage string. “#17VVVVFFFF\r” This will have two possible responses, if the target voltage is different, the HV will ramp to the new value if HV is already on (“ #1603/r” HV ready state) or it will save the new value and continue waiting for the HV on Command (**“>12/r”**)

**Step 3) send a parameters read request, “>05/r” command to verify the current operational parameters**.

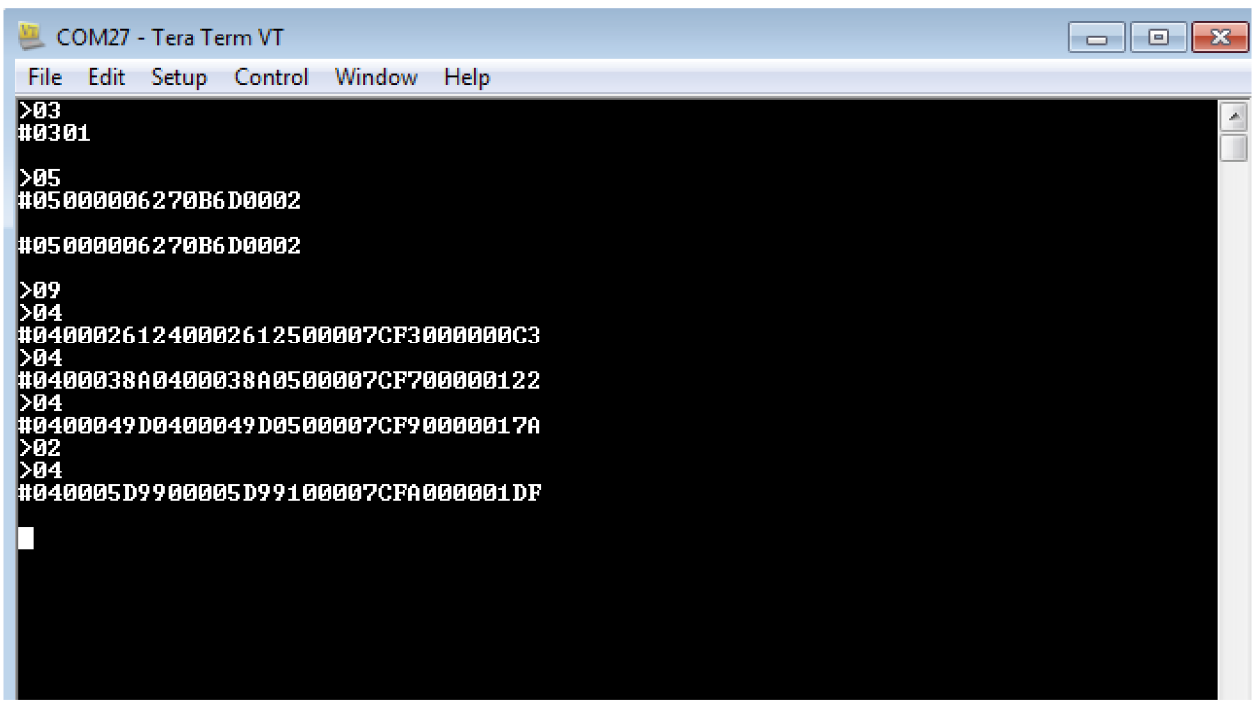
If they have changed since the last sequence, change them by reissuing a new parameter dump: sending a response string back “ #05000006270B6D0002/r” with the desired parameters.

**Step 3: Issue a start count command**
in this example start demo count “**>08/r**” is used.

**Step 4 – N:** Issue sequential request current counts command “**>04/r**” requests to read the counter data.

**Step N + 1:** Stop the counter , by sending a stop command “**>02/r**”

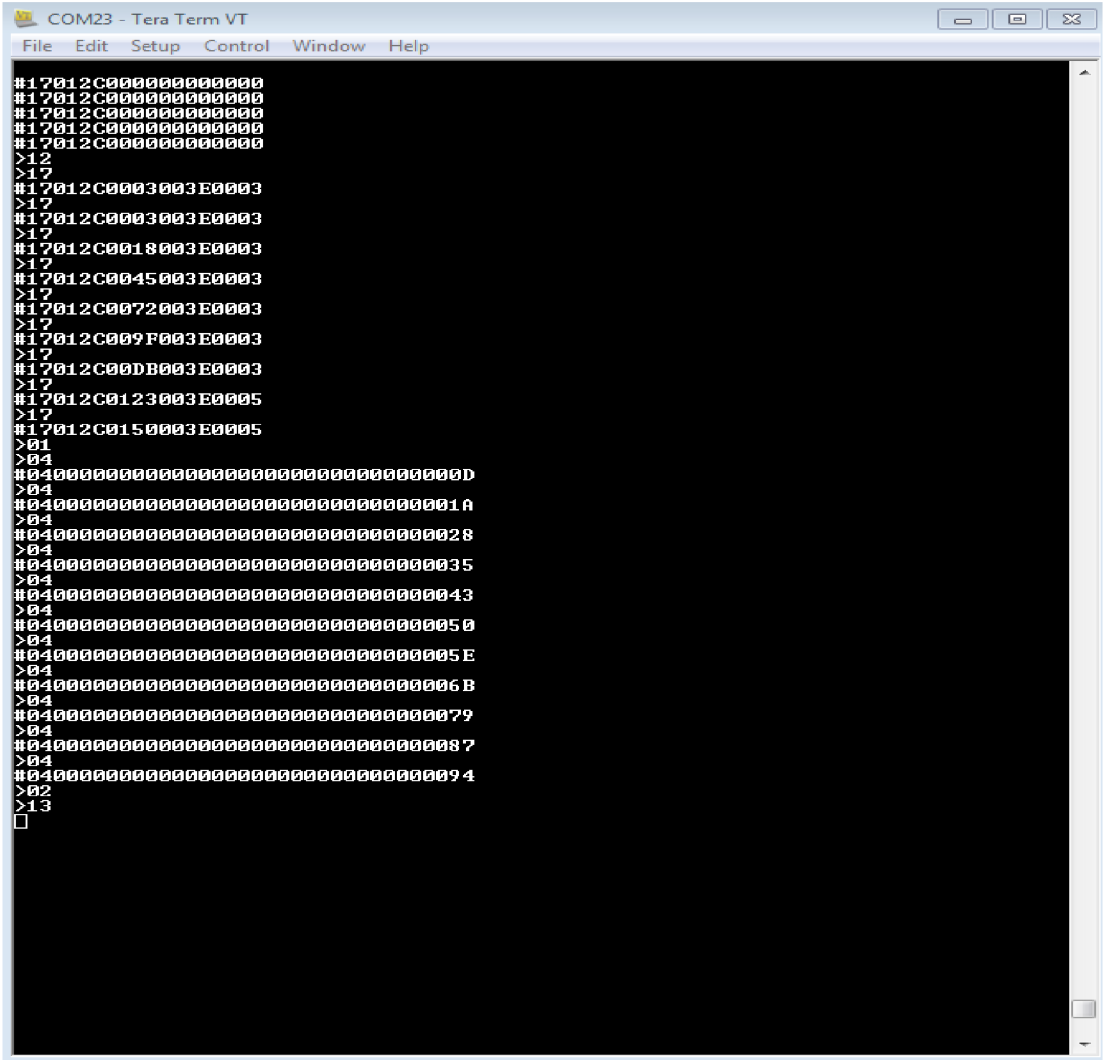
**Step N+2:** Read the final count, after the counters have stopped and been updated by issuing another read counters command “**>04/r**”



## HV off when Start is requested.

This scenario requires that the user checks the HV flags before starting to count. Counting begins when the HV is on the desired voltage and the HV ready flag is set. The following is a screen shot of the communications when the Start button is pressed.

1) The latest parameters are sent to the SCA using the #05….. ack message
2) The latest HV voltage is sent to the SCA using the #17….. ack message
3) A series of read HV data are sent to monitor the flags ">17/r", initially the ENABLE flag is set, "#17 …. 01 " if the 1 wire flag not set . continue to sent ">17/r" till the ramping flag is set "#17…. 03" if the 1 wire flag not set . When the HV has finished ramping and is ready, the HV enable flag will still be set, the HV Ramp flag will clear and the HV_Ready flag will be set . "#17…. 05" .
4) The HV is now settled and the ">01/r" start counting command is sent , data is read with subsequent ">04/r"


COM23 - Tera Term VT
File Edit Setup Control Window Help
#17012C000000000000
#17012C000000000000
#17012C000000000000
#17012C000000000000
#17012C000000000000
>12
>17
#17012C0003003E0003
>17
#17012C0003003E0003
>17
#17012C0018003E0003
>17
#17012C0045003E0003
>17
#17012C0072003E0003
>17
#17012C009F003E0003
>17
#17012C00DB003E0003
>17
#17012C0123003E0005
>17
#17012C0150003E0005
>01
>04
#040000000000000000000000000000000D
>04
#04000000000000000000000000000000001A
>04
#040000000000000000000000000000000028
>04
#040000000000000000000000000000000035
>04
#040000000000000000000000000000000043
>04
#040000000000000000000000000000000050
>04
#04000000000000000000000000000000005E
>04
#04000000000000000000000000000000006B
>04
#040000000000000000000000000000000079
>04
#040000000000000000000000000000000087
>04
#040000000000000000000000000000000094
>02
>13

## STX Hardware Programming

The STX consists of up to 4 modules each with one or more microcontrollers. This document is a step by step guide to programming and verifying each. There are two sections:

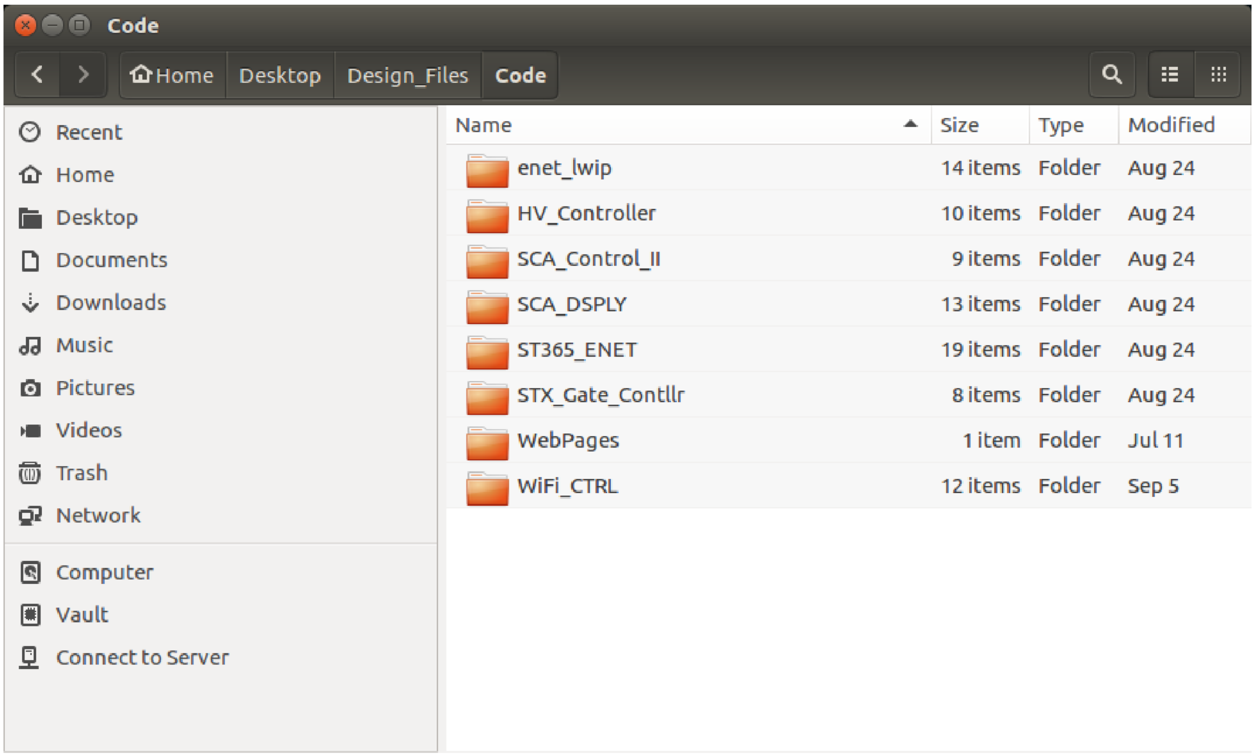
- Firmware development and programming
- Production Programming and test.

The processors used are all in the Texas Instruments family of microcontrollers, the majority are low cost MSP430 type processors. The Ethernet controller and WiFi controller located on the STX Communications board:

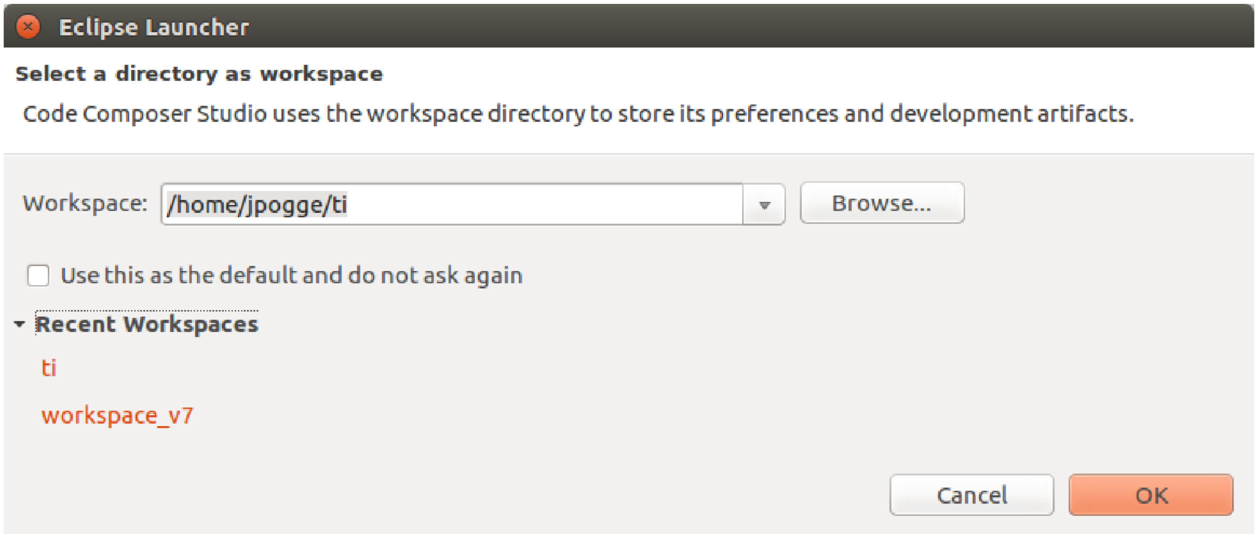
- **ARM® Cortex®-M4 Tiva (TM)** processor and
- **CC3200 ARM A9 WiFi** controller.

## Firmware Development and Programming

Firmware was developed using TI's Code Composer Studio (CCS) tools V7.2. CCS upgrades will require a fresh install to make sure everything is working properly. The Firmware is available for each project board and should be imported as a CCS project to insure the proper settings, configuration and paths are set to allow complete control and proper access to the tools **(Design_Files/Code)**.



Leave the default directory choices of **/home/jpogge/ti** for all add-ons as it is difficult to properly find code and examples otherwise.



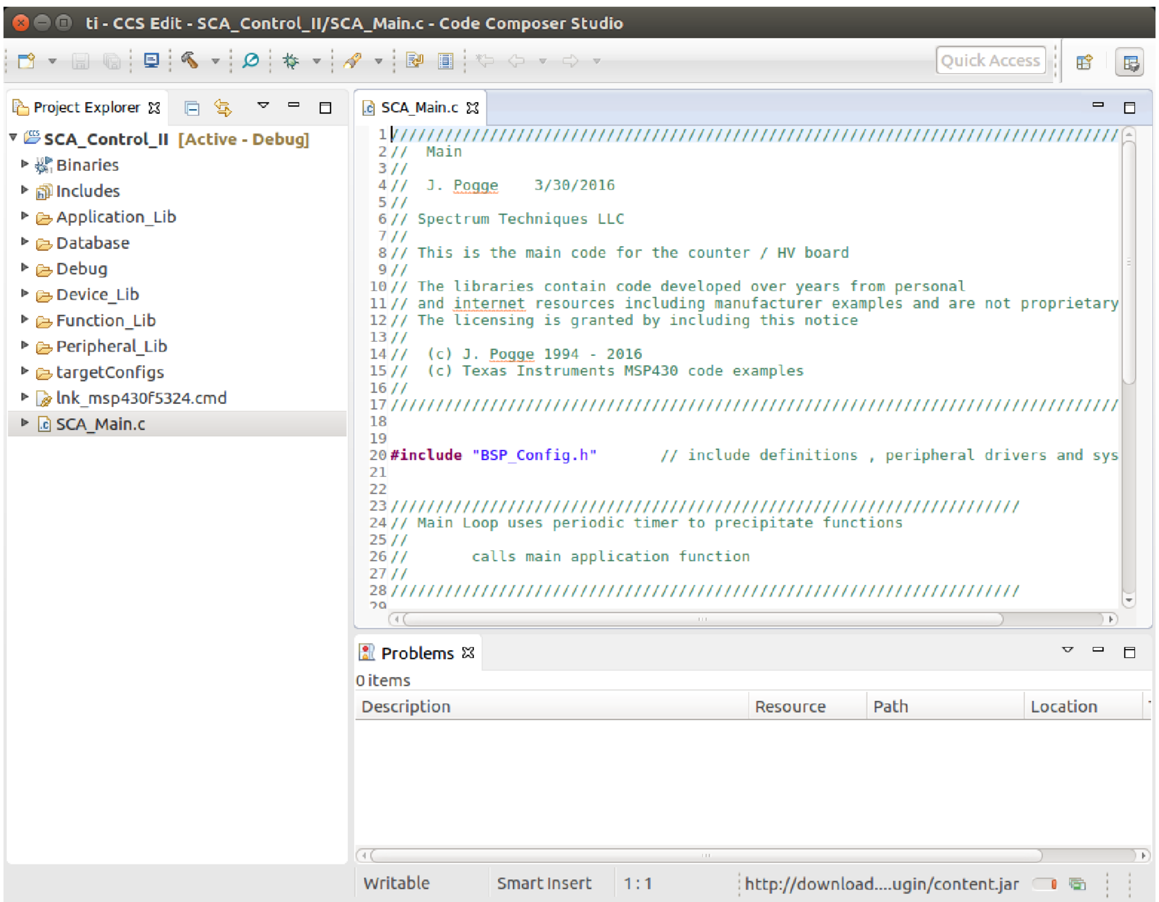

Which results in the following splash screen for the TI CCS version 7.

## Importing a CCS code project.

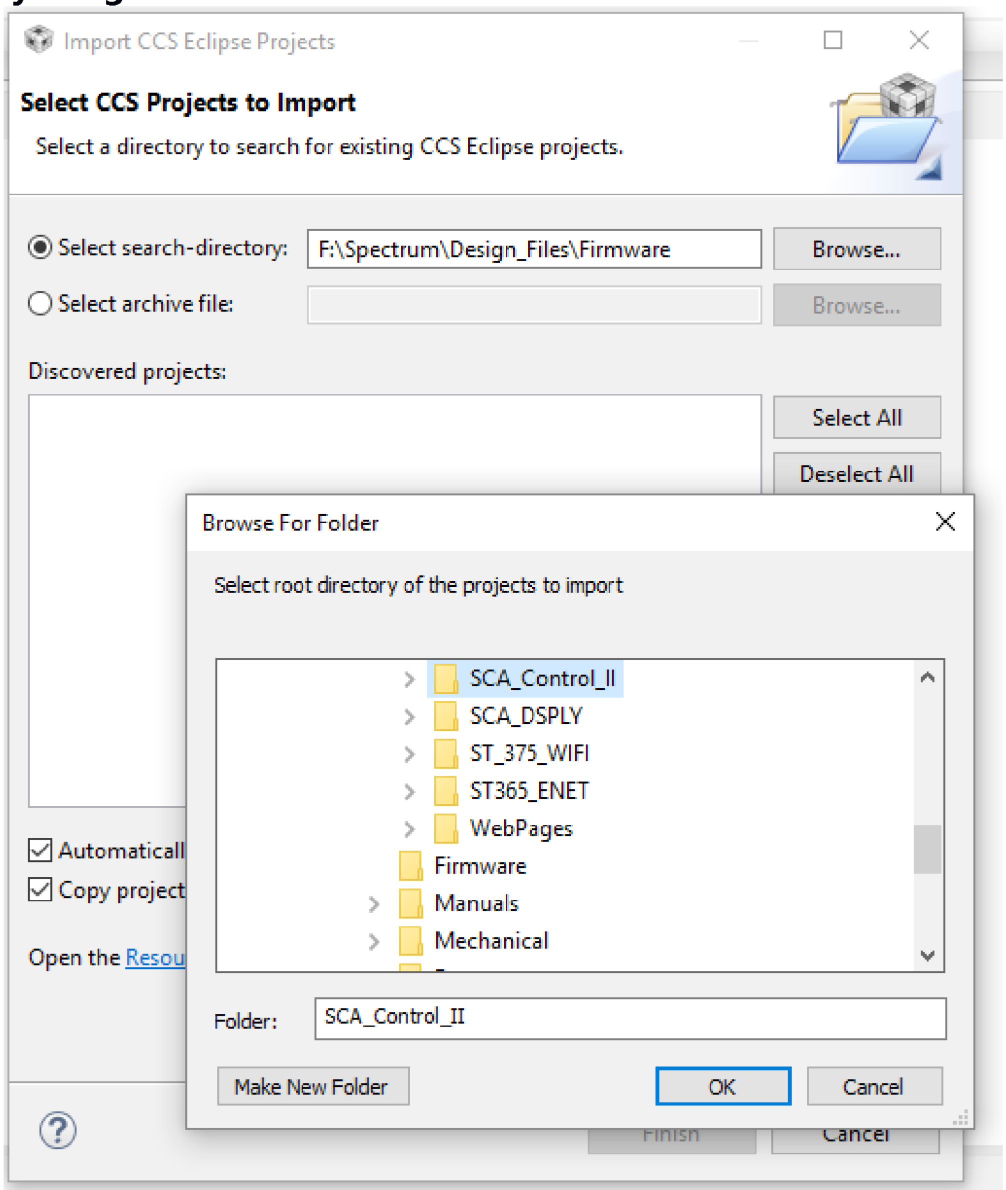
When CCS v7 opens select menu item **Project/Import CCS Project** . Browse to the Code folder located in the Design Folder on the Desktop

### ~/Desktop/Design_Files/Code

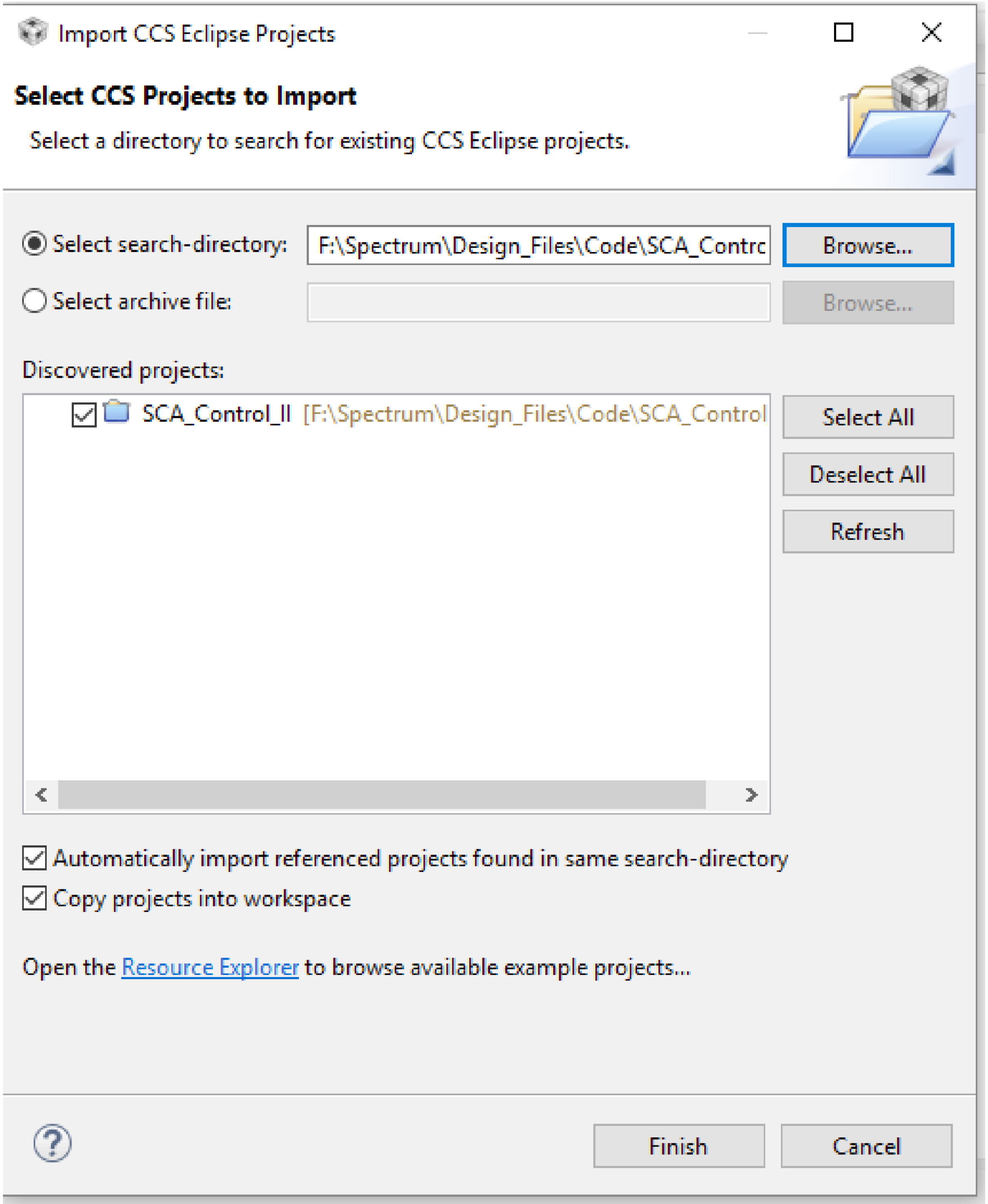
and select the desired project folder, here the SCA Project. The following will appear:



**This is the legacy stuff from Jim that I am saving to ensure that I have everything correct.**




Select the folder only of the project you wish to import, and make sure that the check boxes appear as shown. The project will be rebuilt and installed in your local working directory. See dialog below. Press finish to complete.


Import CCS Eclipse Projects
Select CCS Projects to Import
Select a directory to search for existing CCS Eclipse projects.
Select search-directory:
F:\Spectrum\Design_Files\Code\SCA_Contrc
Browse...
Select archive file:
Browse...
Discovered projects:
SCA_Control_II [F:\Spectrum\Design_Files\Code\SCA_Control
Select All
Deselect All
Refresh
Automatically import referenced projects found in same search-directory
Copy projects into workspace
Open the Resource Explorer to browse available example projects...
Finish
Cancel

## Programming for Production

The production programming is best handled using the **UNIFLASH** tool from TI. It can handle all versions and families of the design. The programming hardware will change depending on the family. All MSP430 STX boards use the **TI MSP430 USB-Debug-Interface MSP-FET430UIF** and the Spectrum Techniques **custom adapter board** shown.

**Figure (1) SCA ready to be programmed.**

1. TI MSP430 USB-Debug-Interface MSP-FET430UIF (dark grey on left).
2. Custom Adapter board (center of photo).
3. SCA Board (DUT) (top).

The custom adapter allows the TI programming device, to connect to the Spy BY Wire (SBW) programming interface on the Device Under Test (DUT). The SBW connection to the DUT is the middle 4-pin Molex connector on the custom adapter board. The SBW is shown in the photo above connecting the custom adapter to the DUT.

The custom adapter also has USB to RS485 for testing all DUTs. The second USB provides power to the DUT. To use the RS485 interface:

1. Connect a USB cable between the custom adapter board and the PC.
2. Connect a RS485 cable between the rightmost 4-pin Molex on the custom adapter board and the RS485 connector on the DUT.

Tera Term VT on Ubunta is used to send ASCII strings to the DUT (in this case a SCA board).

**Figure (2) MSP430 Custom adapter board**

- Connect a four wire JTAG cable between the adapter JTAG port J2 (the white connector in the middle shown above), and the target JTAG as illustrated below. Each of the STX MSP430 boards: the Display Board, HV Board and the SCA Board have a white 4 pin JTAG port on the bottom of the board. Figure (3) shows the JTAG on the HV and SCA boards middle bottom pointed toward inside. And Figure (4) shows the Display Board JTAG on the middle right just above the two pin power connector.

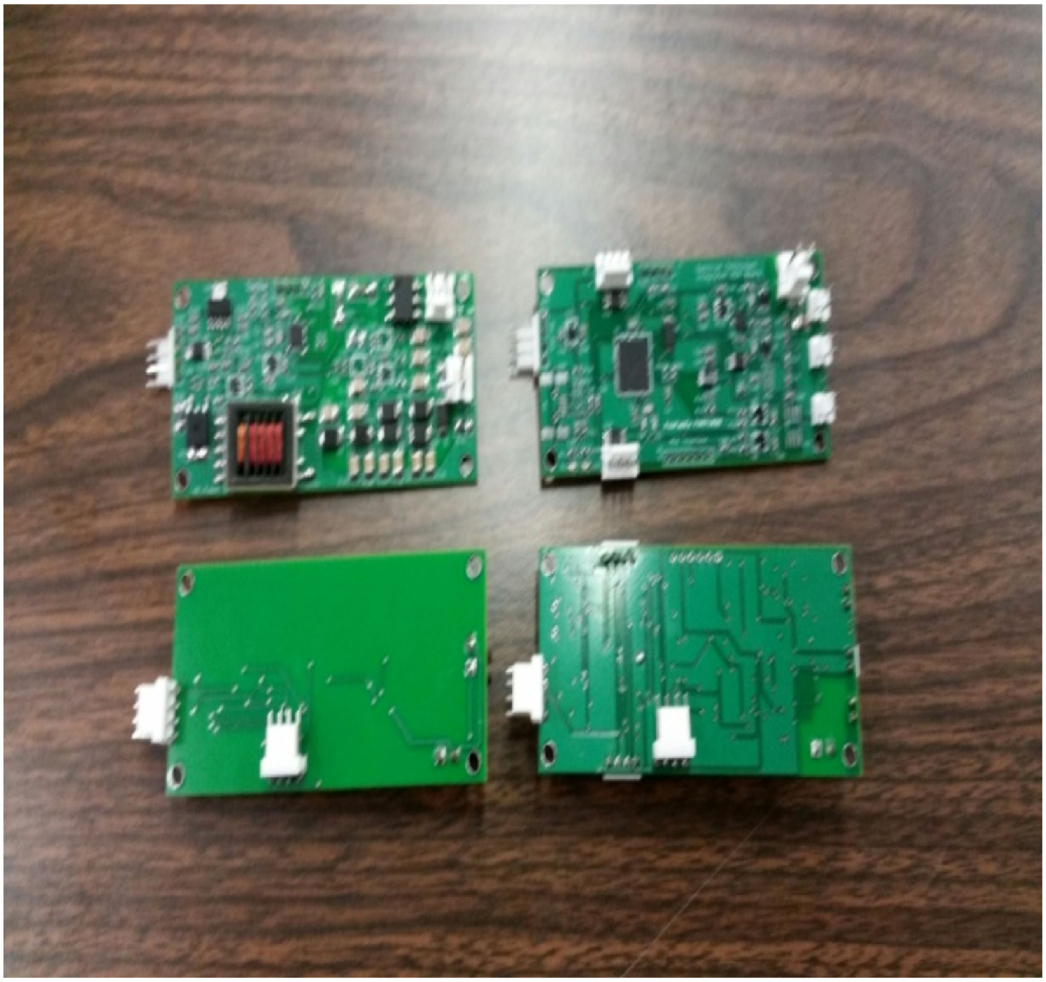


**Figure (3) JTAG on bottom middle of both SCA and HV Boards.**

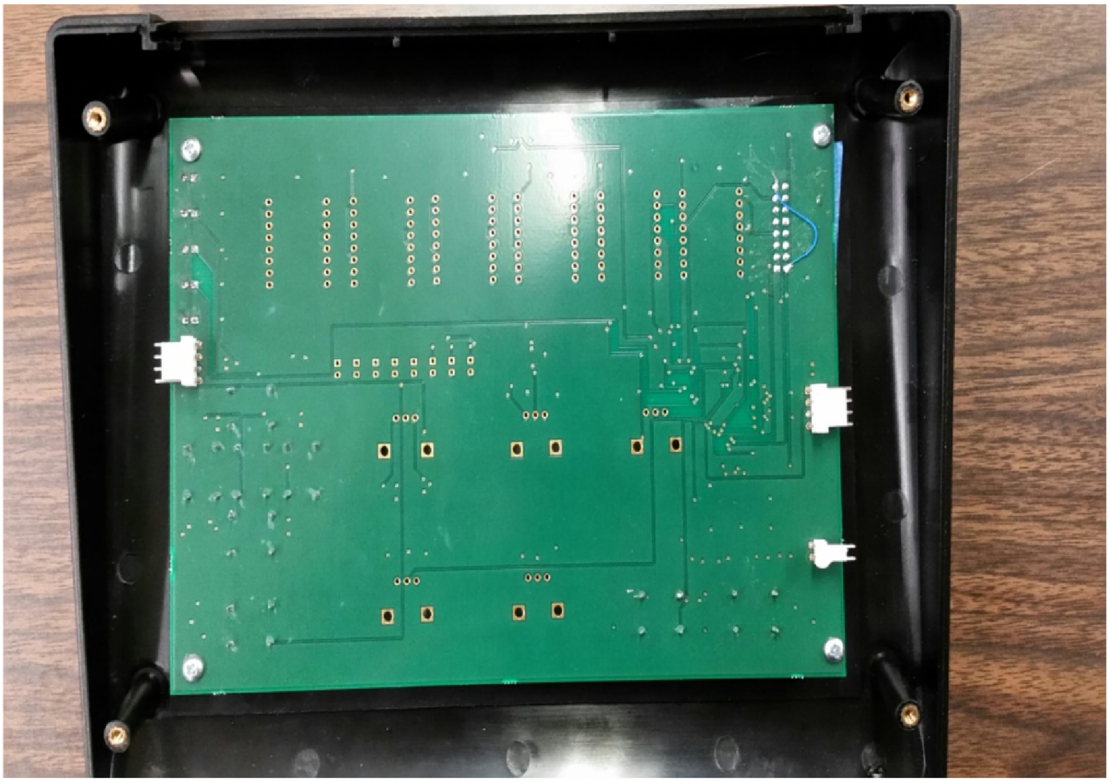


Figure (4) JTAG on middle right above 2 pin power connector.

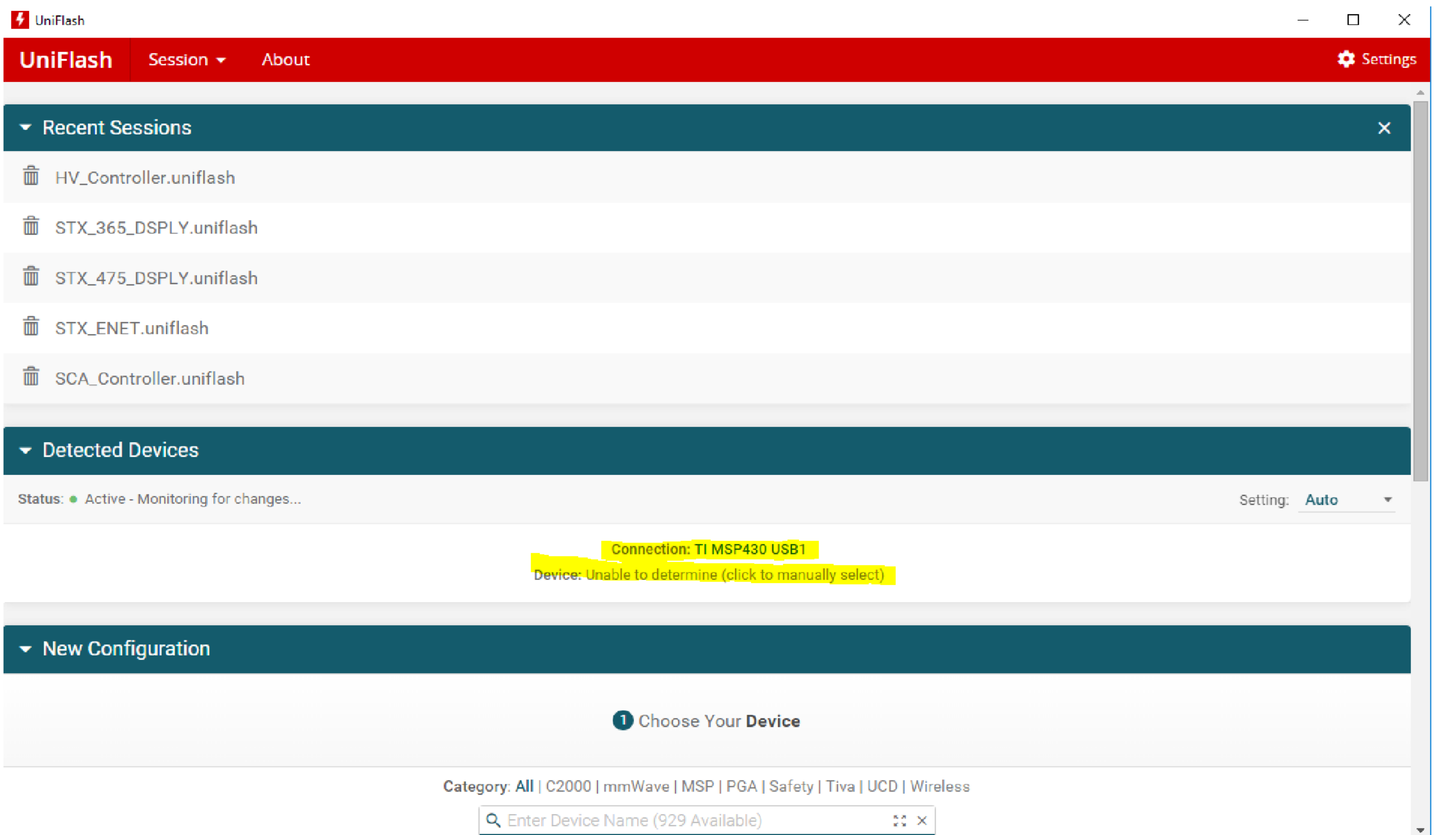
Once the custom adapter board is attached to the DUT, connect the USB to the TI USB programmer, and a second USB to the programming adapter. The second will allow communications to the DUT and provide power during programming and test of DUT.

## Programming with UNIFLASH application software

TI provides a production interface to program all of its devices. The only difference will be the programming adapter. We will focus for now on programming the three MSP430 devices.

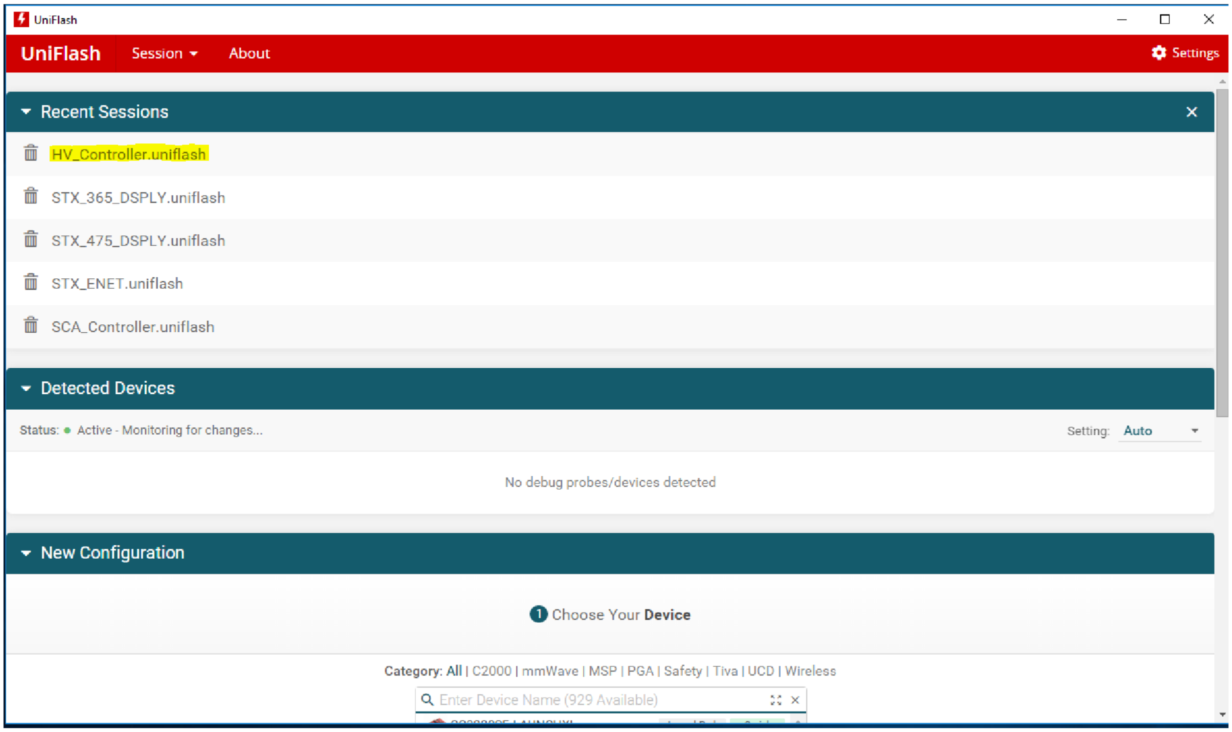
All of the Uniflash scripts are saved and under the main folder usually on desktop **~/Design_Files/Firmware/**

Open the TI Uniflash application.

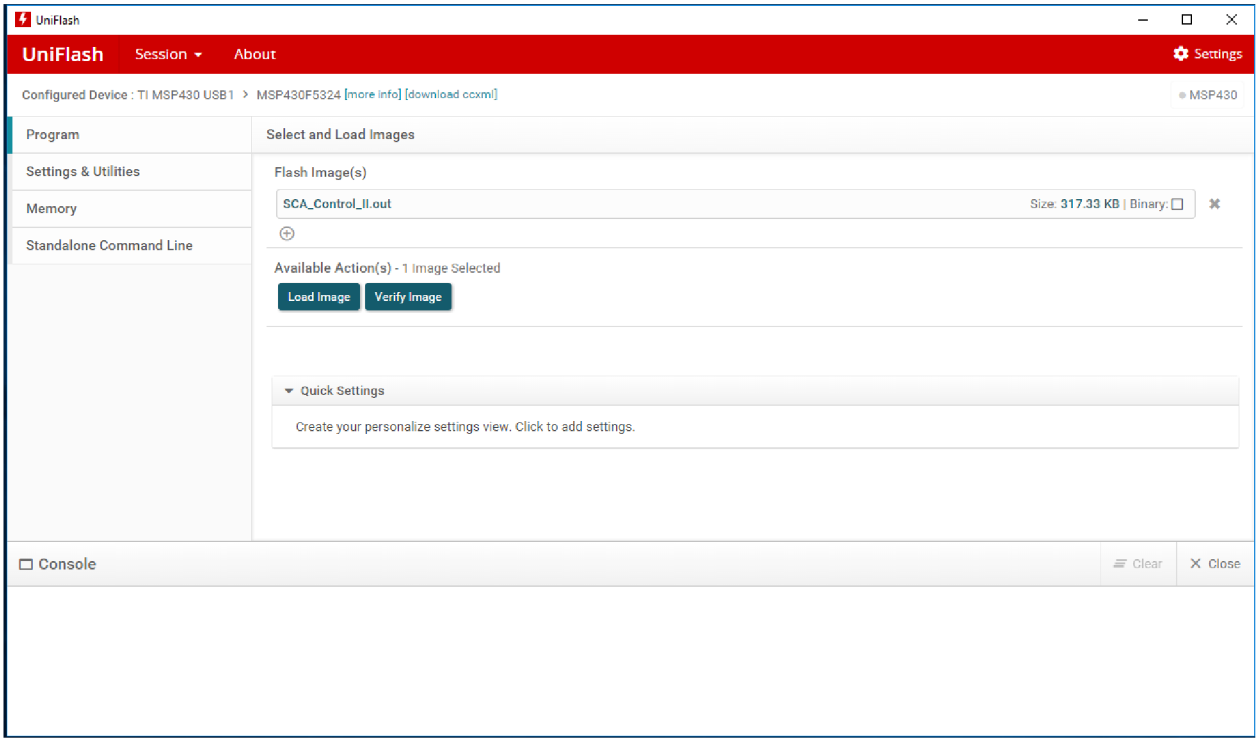



The USB FET programming Adapter is loaded already since it was plugged in, the highlighted connection for the adapter you need must be connected before programming can continue.

Connect the board to the programmer and select the Uniflash script from either the recently used or from the Firmware directory under the **~/Design_Files/Firmware/** folder.




In this case the highlighted file indicates that we will program the HV controller board. Once it is loaded a similar screen will appear as below ;




When the target board is connected press the button **Load Image**. The Uniflash software will indicate either success or a problem with the connection of the board.

If this is a new board two things stand out with MSP430 and most TI parts.

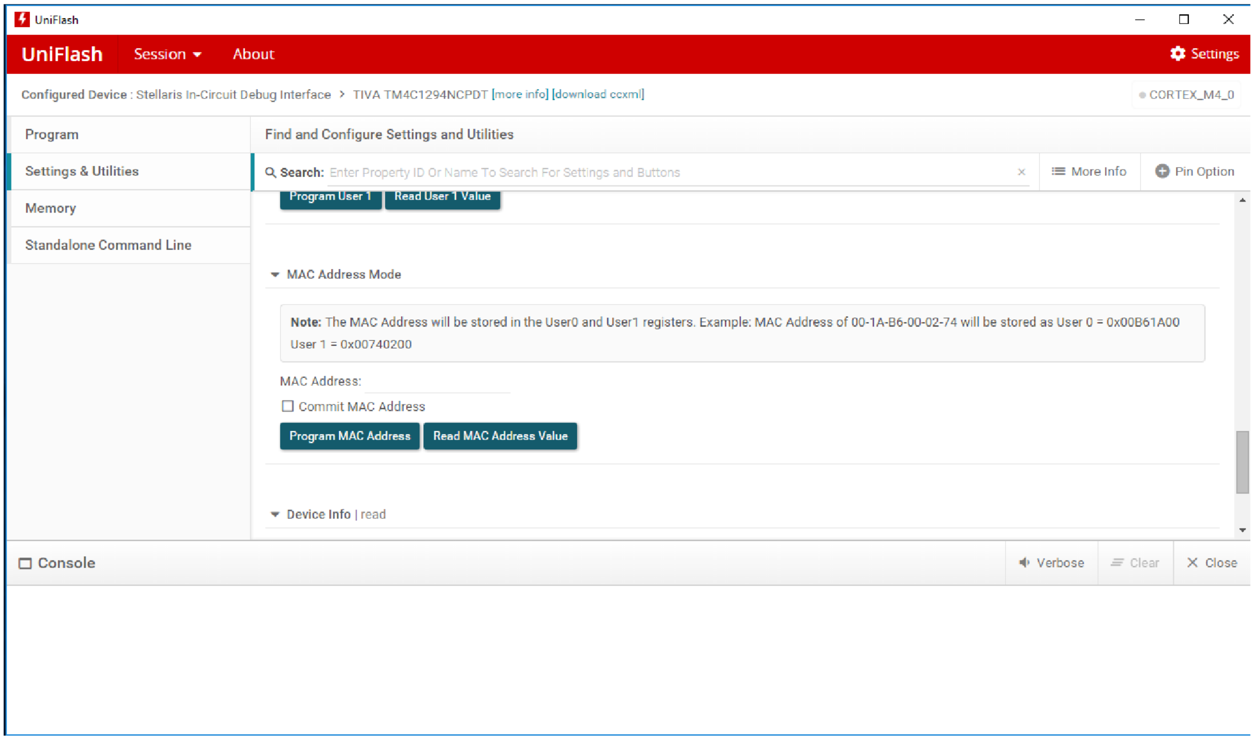
1. Even though the data sheet says the ground pad isn't critical, it must be connected to ground for the programming JTAG port to work.
2. The reset line R/C must have a 2.2nF (2200pF) cap to ground and a 4.7K pull up resistor. Other values will mess with the JTAG signals

If the board does not program even after three retries, then either the processor is damaged, or there is a solder connection or loose wire in the JTAG chain.

The CC3200 requires the previous version of **Uniflash 3.14.1** to work at the time of this writing. The above examples use **Uniflash v4.** (8/24/17)

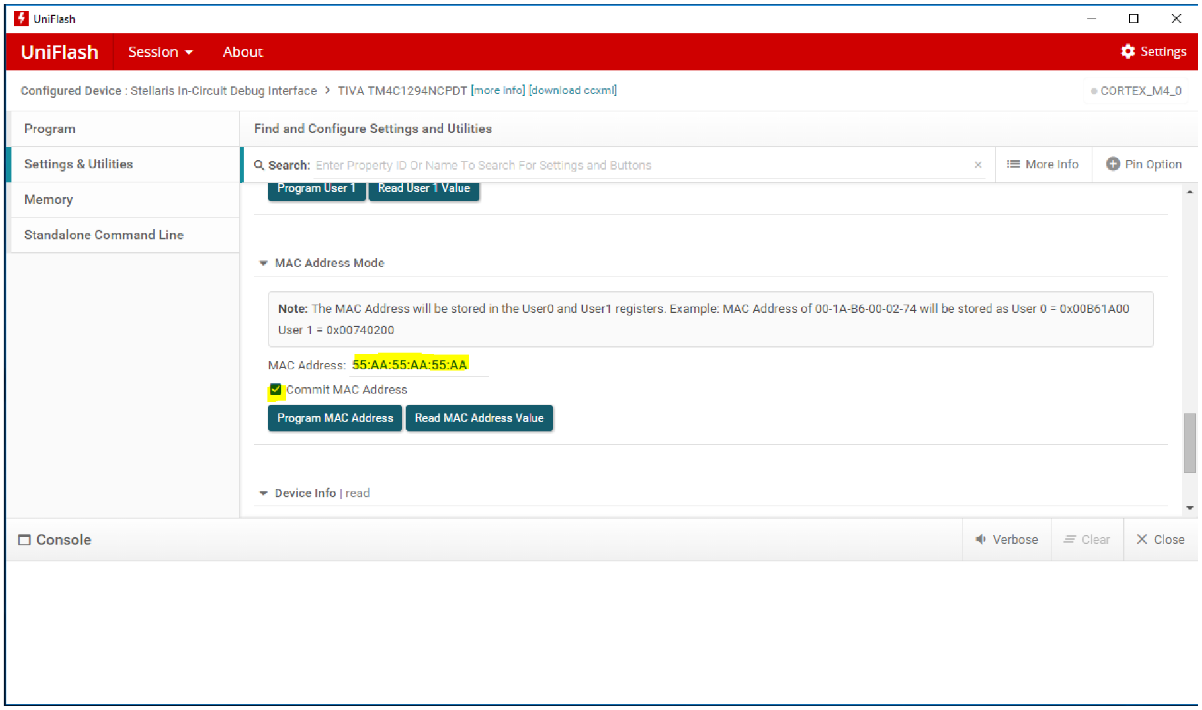
When programming an Ethernet Comm board, an additional step is required. The Adapter must be the **TI XDS100V2 type ARM programmer** , (Not the MSP430 adapter) and after the device has the firmware loaded a **second step is required to program the Ethernet MAC address**.

1) Select the Settings and utilities dialog as shown in the next image. And scroll down to the MAC programming section.




Spectrum Techniques has ordered 4096 unique MAC addresses from IEEE , see attached email for instructions for getting a new MAC in the Appendix.

2) Request a new MAC address from IEEE and save it. This must be unique to this board and can only be used on this board. If several boards are to be programmed make a multiple request for MAC addresses from the IEEE website to have one for each board.
3) Enter the new MAC address as capital HEX numbers . There should be six pairs of numbers (48 bits), and they must be entered on the line with a colon between each as shown.
4) Make sure the "**Commit MAC Address**" box is checked as shown.
5) Press "**Program MAC**" button




If a programmed board does not boot and blink the Ethernet LEDs when connected, repeat the previous steps. The code firmware will freeze if the MAC address detected is invalid, so no violation of Ethernet protocol takes place.

# Schematics

The schematic capture program for the PCB design of the boards is **Altium V17**.

## Schematic Organization

This section has a copy of each schematic. The design is not contained on a single page, but rather each page represents a specific design feature: CPU, USB, Connectors, Amplifier etc..

## Network Naming Convention

The convention to follow is signal names or wire list (nets) for network list contain unique signal names to differentiate the connection. Even if no physical line can be seen connecting two wires on a schematic, if they share the same NET name the schematic capture will connect them as one.

This method eliminates the long winding wires on a schematic that must be followed carefully to see if a connection exists.

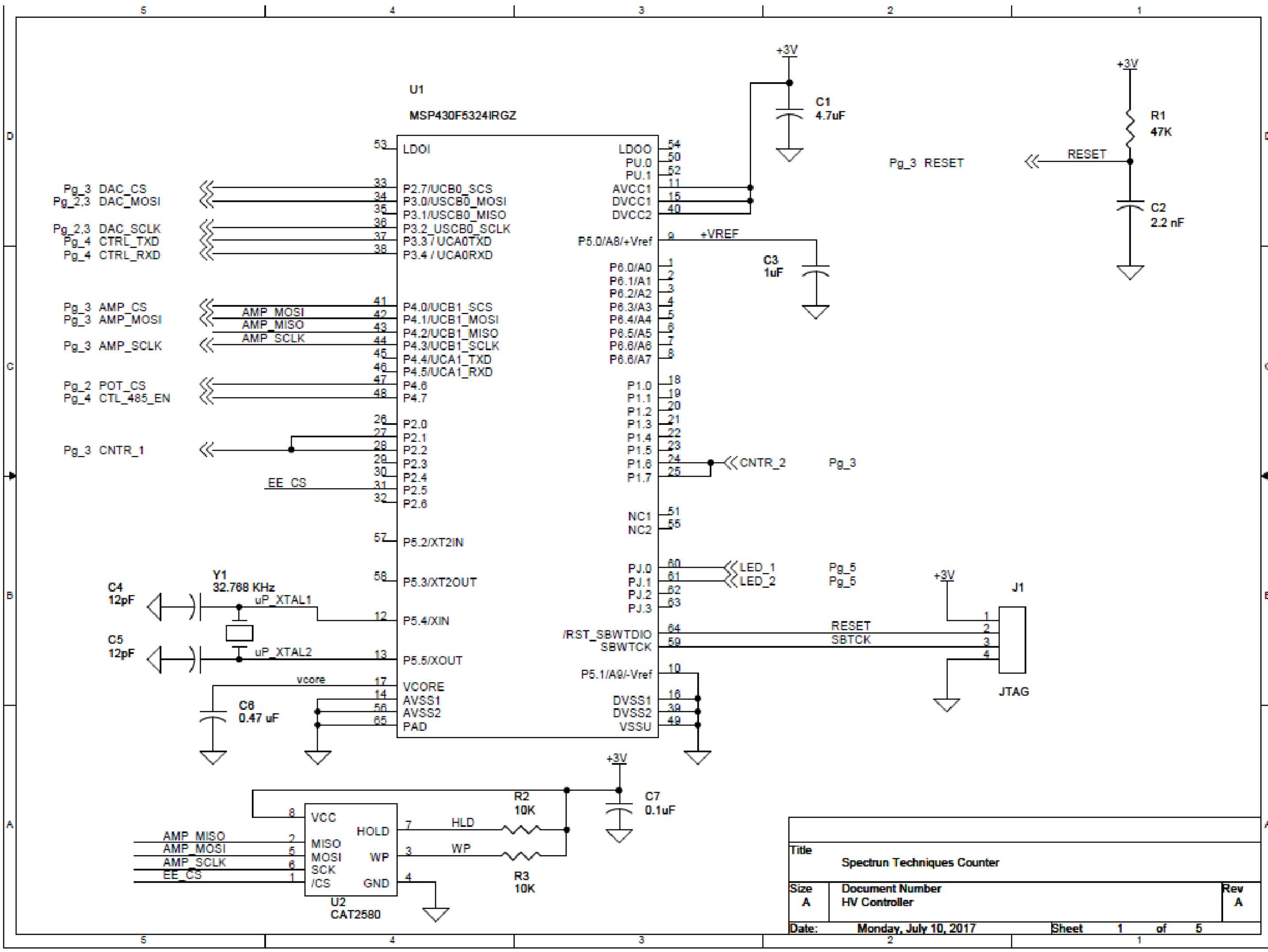
All of the schematics share the following design rules.

1) Ground symbols that look alike are all connected together.
2) Power symbols have a T bar and the corresponding voltage on top, the label must match for a connection to be made by the software. This includes upper and lower case characters. 3v and 3V, and 3.0V will not be connected together on any of the schematics, so these have been checked for proper use of the net names.
3) Off page net connectors have the NET name and page numbers where the share net will be found. Off page connectors all use >> or << line endings to indicate the connection goes to another page.
4) If a net (wire) stub exists , one with no off page connector symbol and a net name but only one side connected. This indicates that one or more similarly named net stubs will be on the same page. All net (wire) stubs with the same name will be physically connected on the PCB.
5) The schematic has the symbol for the device used such as resistor , capicator, IC (integrated circuit) and the value 4K , 10uF, SN75HC74, but not the package type. Package and footprint appear in the corresponding BOM (Bill of Materials) parts list and the PCB design file. Adding package makes the schematic hard to read and does not convey useful info to the technician during trouble shooting.

# SCA Schematics

## SCA Microcontroller

The TI MSP430F5324 microcontroller consists of the main CPU and peripherals (clocks, counters, timers, serial ports, ADC, parallel ports (GPIO), etc), memory and the programming JTAG interface.



### SCA Amplifiers

Both the One-wire signal and the Two-wire signal are present on the SCA. J2 has the 0ne-wire and J3 has the Two-wire labeled “DET_IN”. The signals are both present at SIG_IN and are presented to both operational amplifiers U4 and U5.

The Two-wire signal from the radiation detector has had the HV blocked using a 470pF 2KV blocking capacitor, C14.

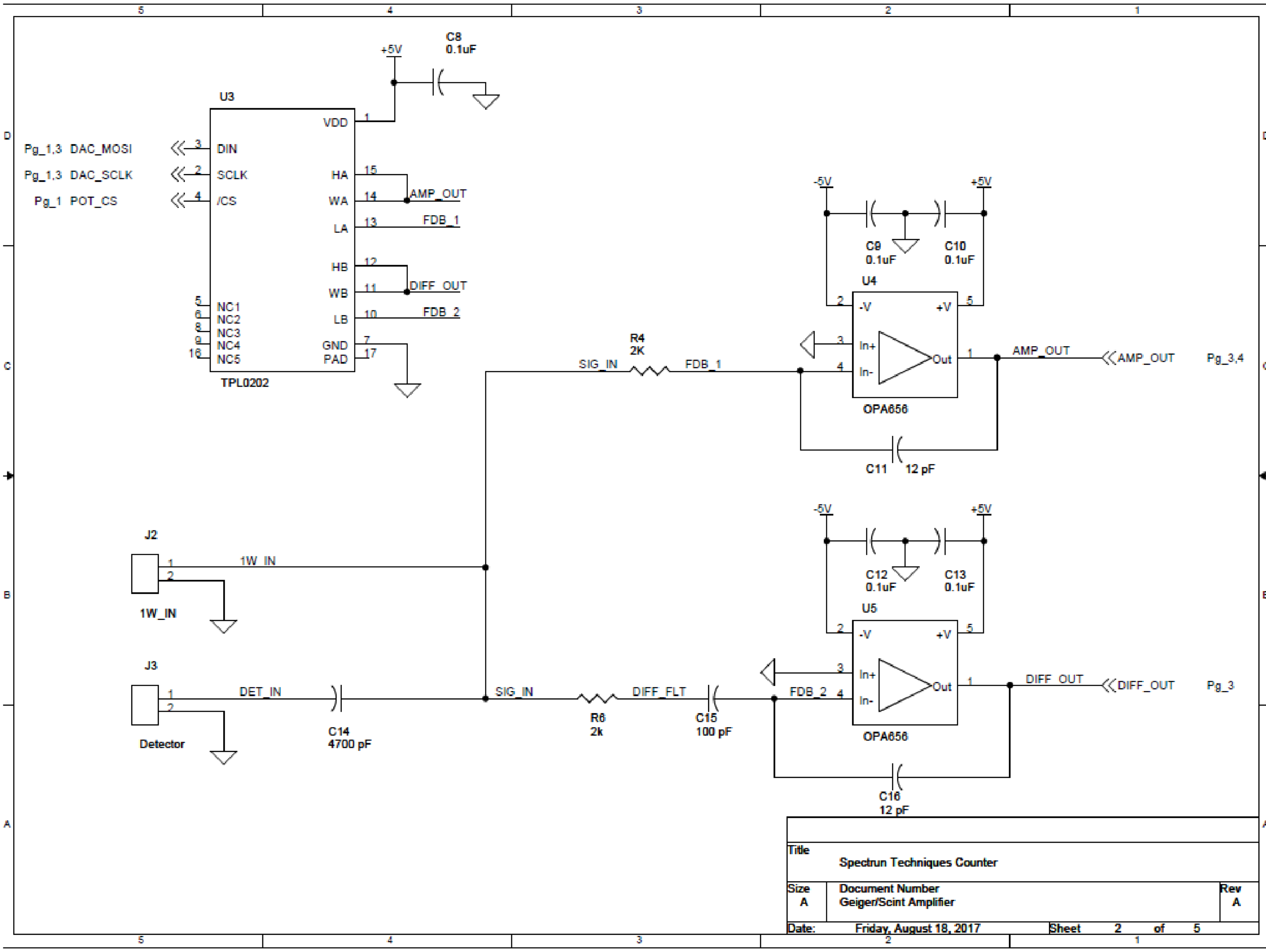
The original signal is a negative semi Gaussian pulse formed by the charge transfer from the detector electronics. Both amplifiers provide an inverting function.

The operator selects between two signal paths:

1) An inverting operational amplifier (U4) used for the G-M tube.
2) The differentiator operational amplifier (U5) for the scintillation detector.

   The differentiator operational amplifier is used to eliminate the long tail caused by slow capacitance discharge to avoid stack up. The output of the differentiator is higher than the original pulse because the area under the curve representing the total power (charge) transfer is equivalent.

   A programmable resistor chip, U3, sets the gain for the selected operational amplifier U4 or U5. The microcontroller selects a gain of 1,2,4,5,8,10,16, and 32 via the SPI interface.



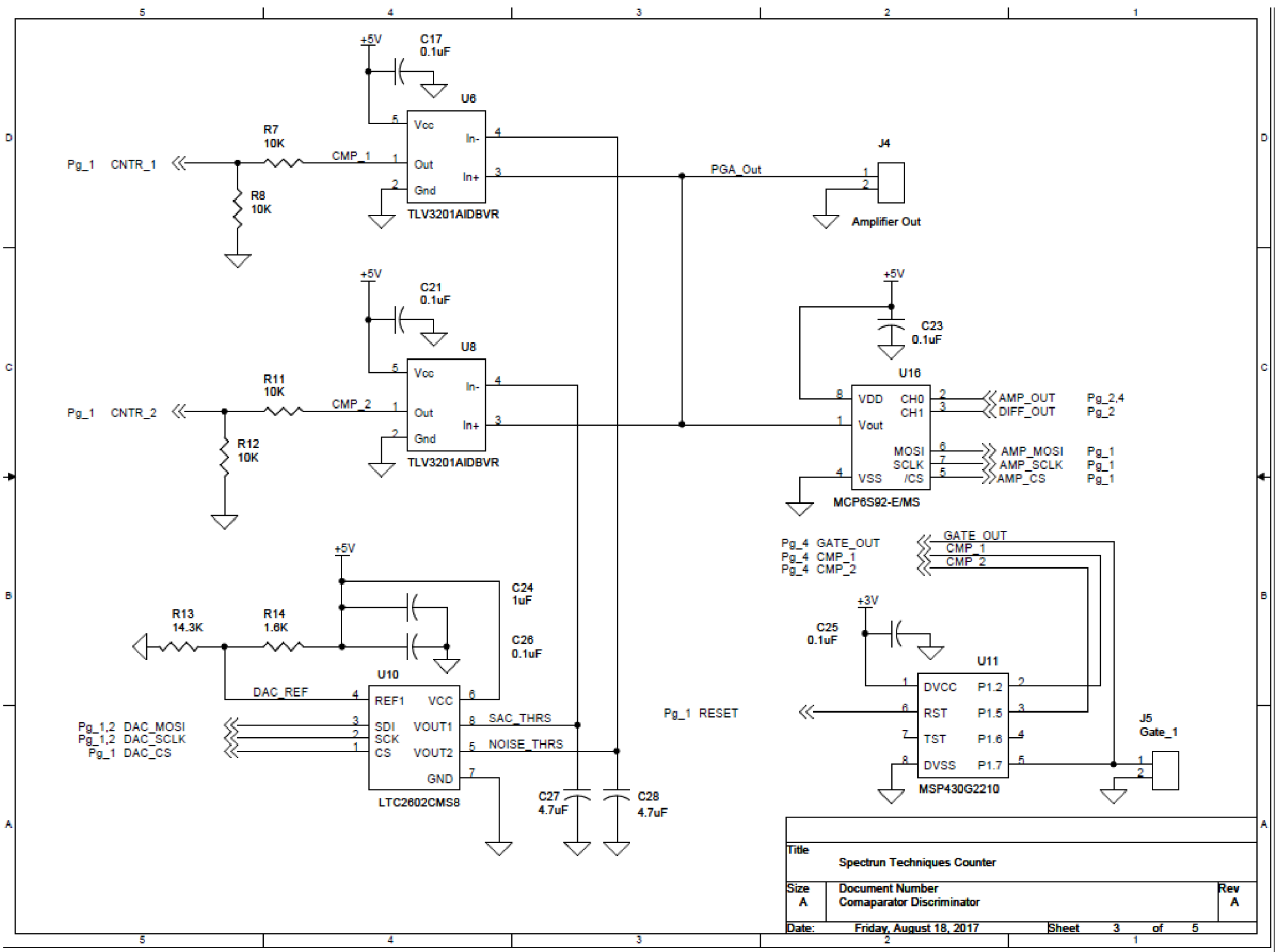
C8
0.1uF
+5V
U3
VDD
Pg_1,3 DAC_MOSI
Pg_1,3 DAC_SCLK
Pg_1 POT_CS
DIN
SCLK
/CS
HA
WA
LA
HB
WB
LB
GND
PAD
NC1
NC2
NC3
NC4
NC5
AMP_OUT
FDB_1
DIFF_OUT
FDB_2
TPL0202
R4
2K
SIG_IN
-5V
+5V
C9
0.1uF
C10
0.1uF
U4
-V
+V
In+
In-
Out
OPA656
AMP_OUT
Pg_3,4
C11
12 pF
J2
1W_IN
J3
DET_IN
Detector
C14
4700 pF
R6
2k
DIFF_FLT
C15
100 pF
FDB_2
C12
0.1uF
C13
0.1uF
U5
DIFF_OUT
Pg_3
C16
12 pF
Title
Spectrun Techniques Counter
Size
A
Document Number
Geiger/Scint Amplifier
Rev
A
Date: Friday, August 18, 2017
Sheet 2 of 5

## SCA comparators

The operational amplifier outputs, AMP_OUT and DIFF_OUT are fed to a multiplxer, U16. The microcontroller selects the output of U13 via the SPI interface. U13's output is fed to two comparators. A 16-bit DAC provides the threshold voltage used in the comparisons.

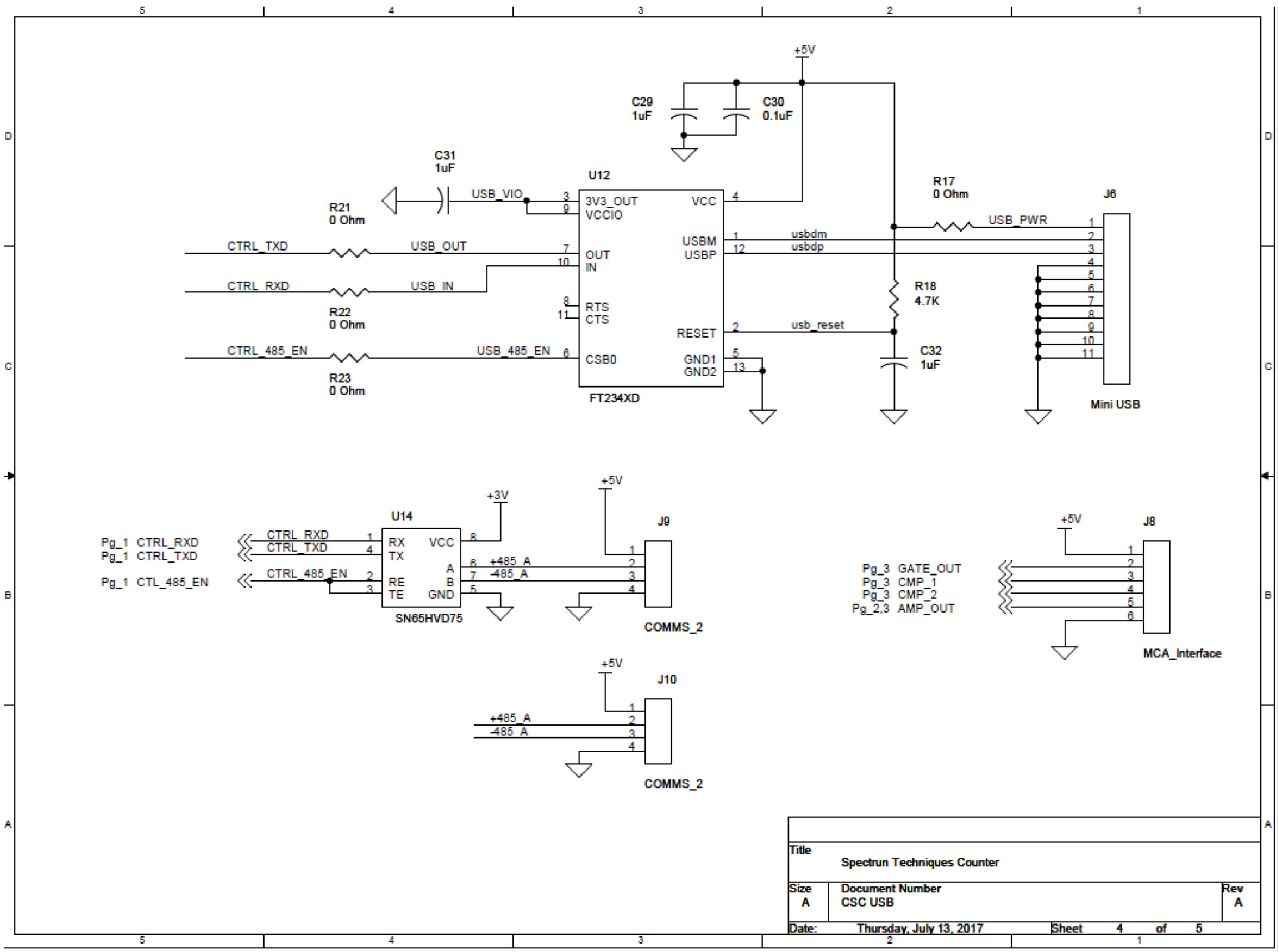
The comparator output pulse is directly connected to both of the register counter input clocks on the second microcontroller, U11 (MSP430G2210). Counter 1 counts all pulses larger than the low threshold (100mV), and counter 2 counts all pulses larger than the window threshold (4.2V).

**Need an explanation of how the 2nd uC handles the counting function and sends the results to the main SCA uC. How is J8 used MCA intrf?**



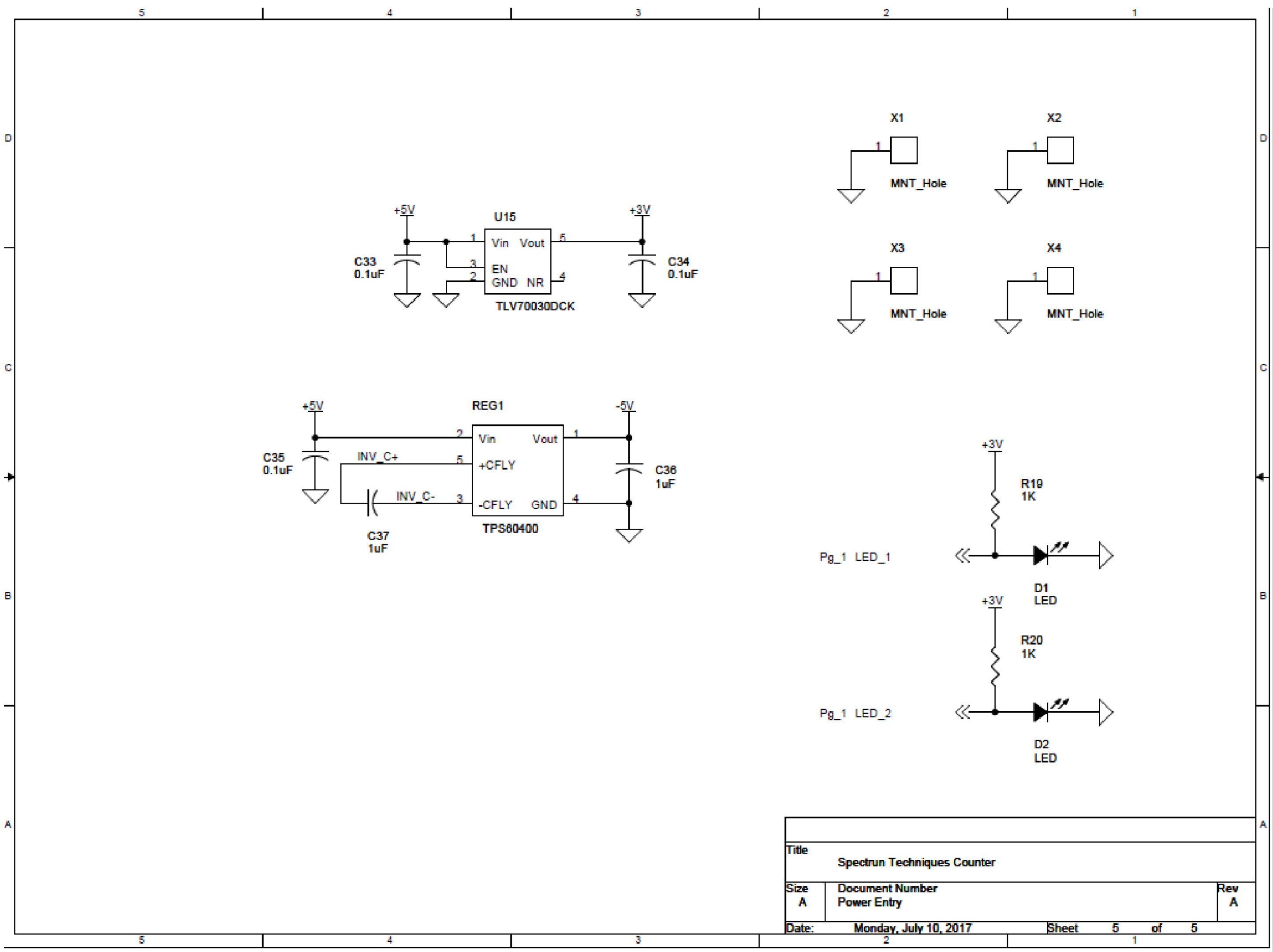
## SCA Communications

This schematic contains the RS485 translators, and the USB to serial converter. As mentioned the external 5VDC regulated power input is on any of the RS485 connector ports , pin 1. **There is no USB on the SCA Board??? The SCA Board in the lab has 3 RS485 ports on it???**



## SCA Power Regulators

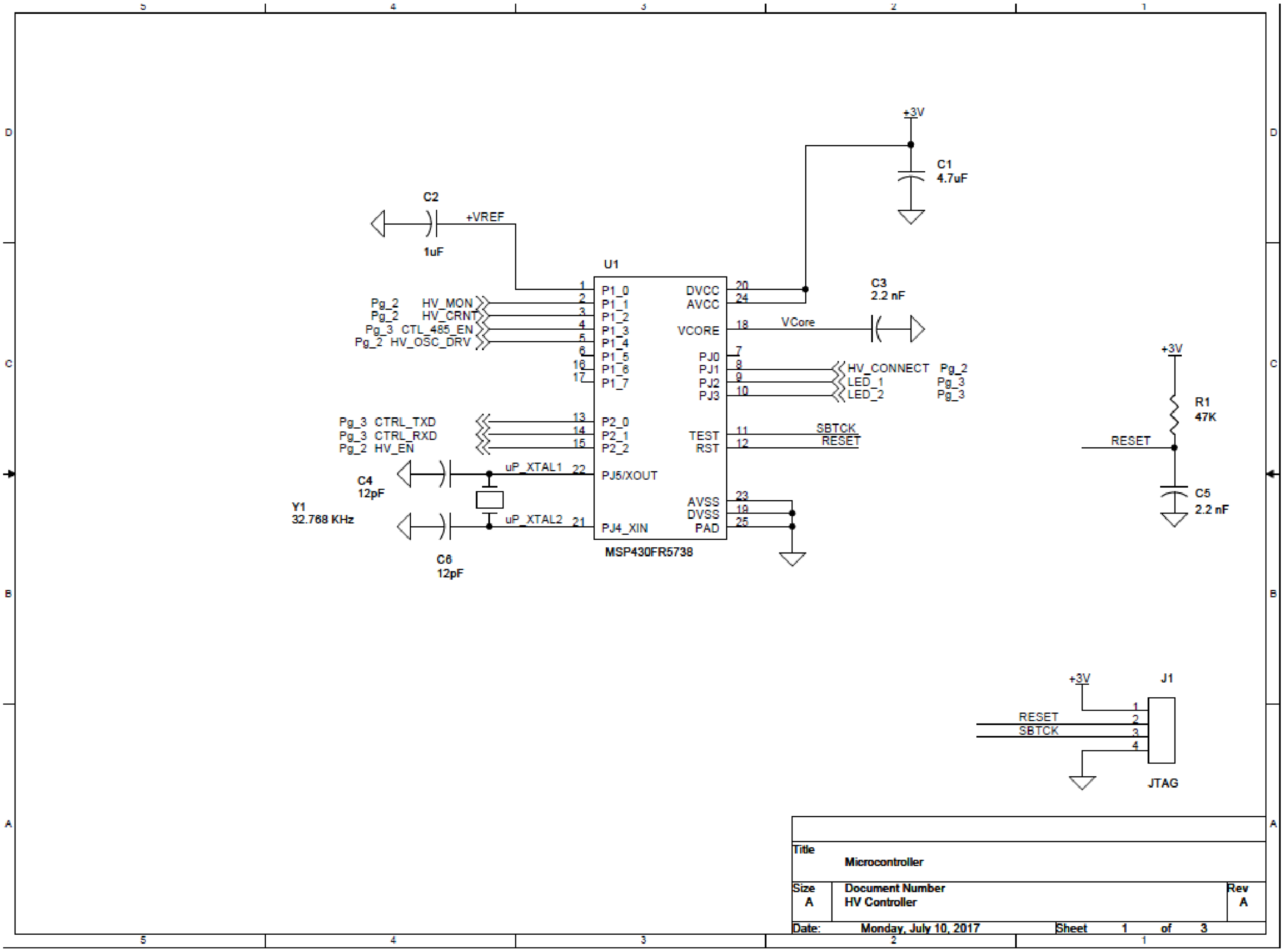
The 5VDC input is regulated and used to provide local microcontroller 3VDC power. Additionally, it is inverted using a charge pump inverter providing 100mA of -5VDC for the analog electronics interface. The -5VDC is used as the negative rail for the two operational amplifiers. No negative voltage are used with the uCs. The uCs use 3VDC for analog and digital VCCs and GND for all VSS. **The internal reference for the analog ADC and DAC in the uC is assumed to be 2.5V.**



# HV Schematics

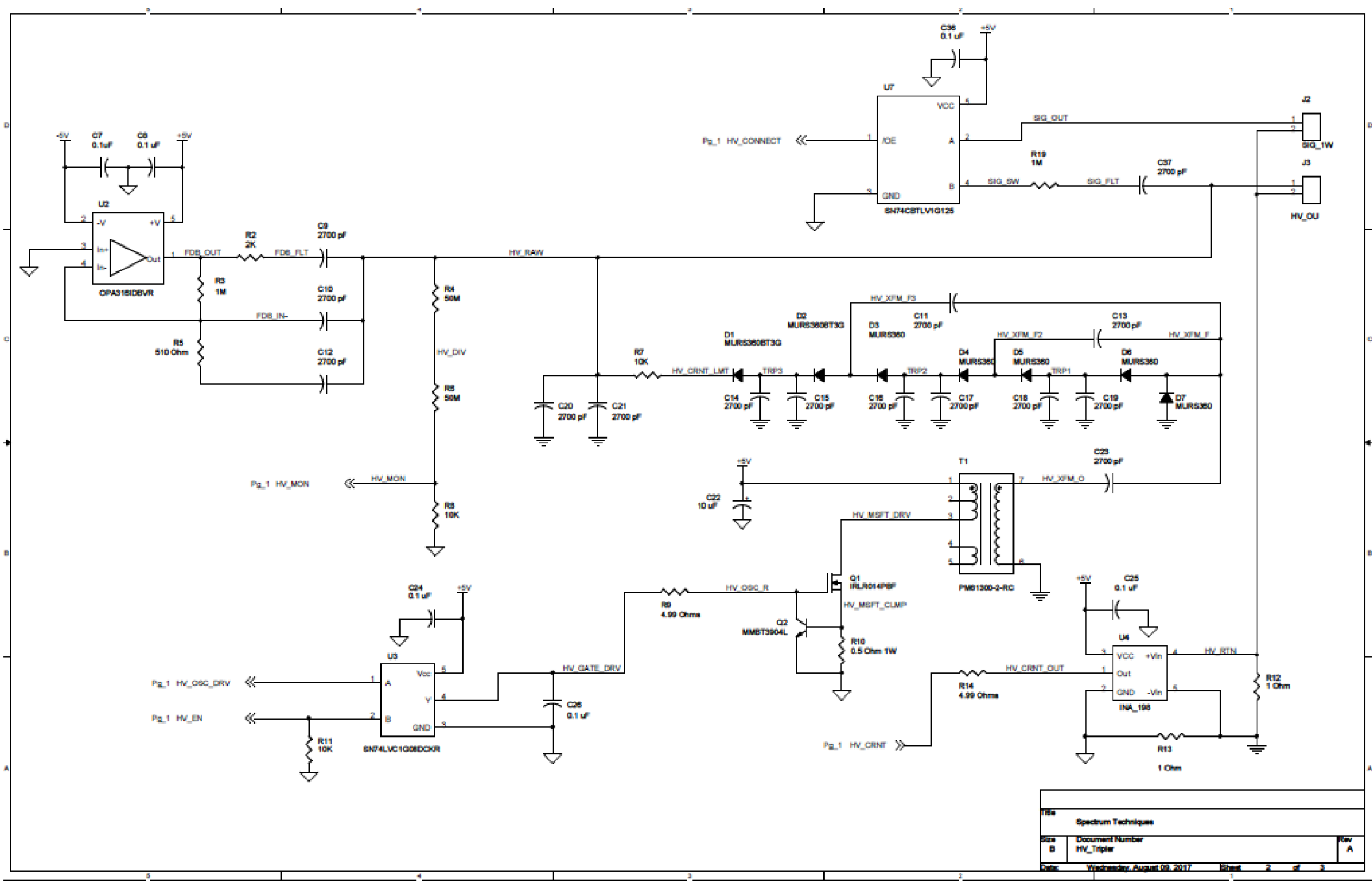
## HV Microcontroller

The HV microcontroller is isolated through the transformer from the high voltage and a common ground link between digital ground and high voltage ground is provided through a 1 Ohm power resistor. This HV Board microcontroller is a TI MSP430FR5738. The firmware is designed to prevent the HV Board from booting up hot (HV ON). A reset or restart will always come up in idle mode waiting for a HV ON command.


+3V
C1
4.7uF
C2
+VREF
1uF
U1
P1_0
P1_1
P1_2
P1_3
P1_4
P1_5
P1_6
P1_7
DVCC
AVCC
VCORE
C3
2.2 nF
VCore
Pg_2 HV_MON
Pg_2 HV_CRNT
Pg_3 CTL_485_EN
Pg_2 HV_OSC_DRV
PJ0
PJ1
PJ2
PJ3
HV_CONNECT Pg_2
LED_1 Pg_3
LED_2 Pg_3
R1
47K
RESET
Pg_3 CTRL_TXD
Pg_3 CTRL_RXD
Pg_2 HV_EN
P2_0
P2_1
P2_2
TEST
RST
SBTCK
RESET
uP_XTAL1
PJ5/XOUT
C4
12pF
Y1
32.768 KHz
uP_XTAL2
PJ4_XIN
AVSS
DVSS
PAD
C5
2.2 nF
MSP430FR5738
C6
12pF
+3V
J1
RESET
SBTCK
JTAG
Title
Microcontroller
Size
A
Document Number
HV Controller
Rev
A
Date:
Monday, July 10, 2017
Sheet 1 of 3

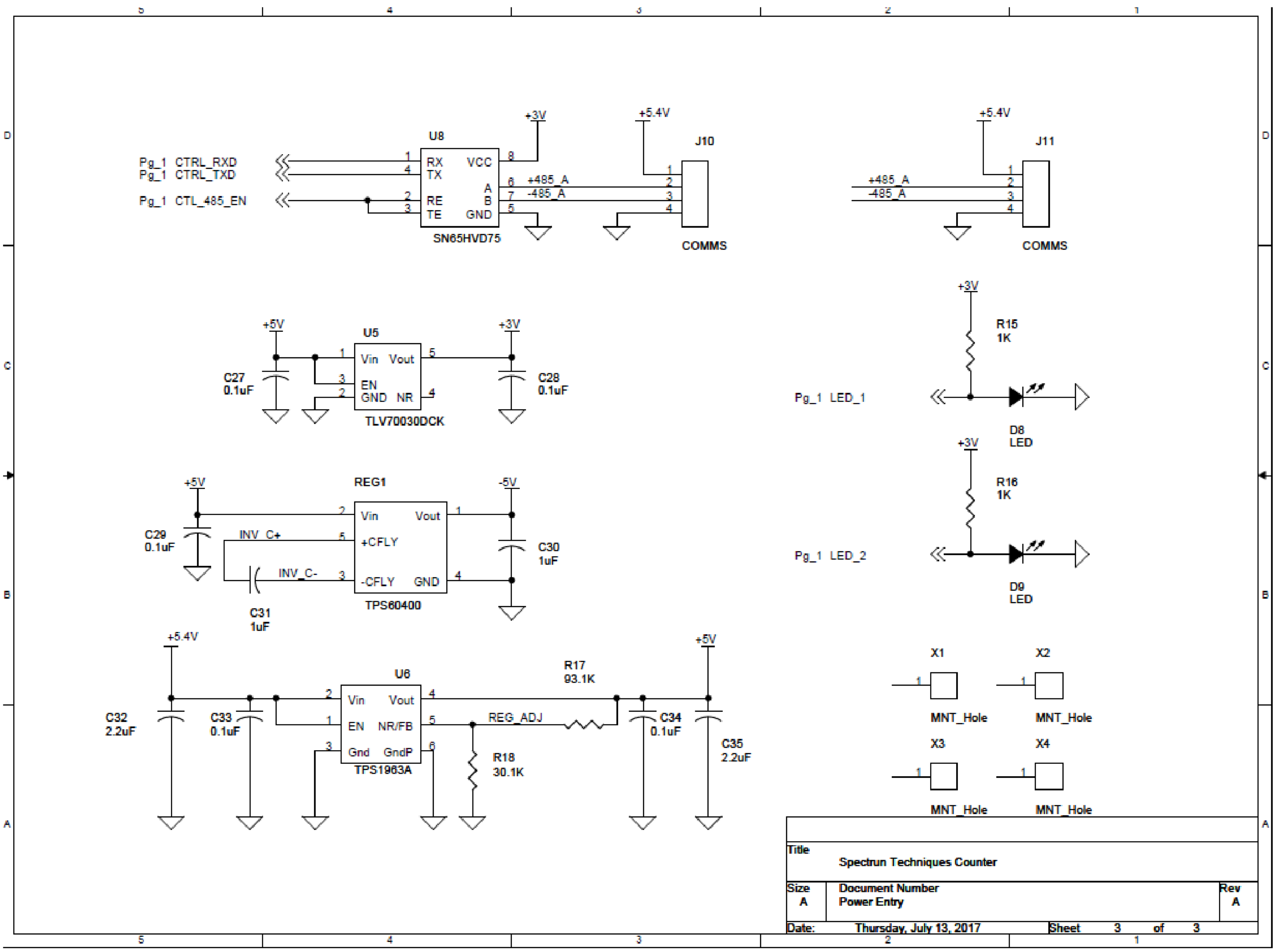
## HV Power Amplifier

This design uses a step up transformer 120:1 and a three stage HV amplifier pump also called a Tripler. This allows the controller to switch a signal 5V PWM (pulse width modulated) through the 120:1 transformer to producing an approximate voltage gain of 600V. The Triplet doubles this voltage by pumping through cascaded diode stages into charging capacitors. This type of amplifier is intrinsically noisy. The PWM freq is set to 30 KHZ which is ½ the fundamental resonance of the transformer. The PWM should not be switch much higher than 50% duty cycle as this will cause heating of the transformer coil. The voltage output of the amplifier is set by adjusting the pulse width of the PWM signal to the transformer. The expected load is 4-6 M ohms. Operating above or below this load will not work for the 100Vdc – 1200 V range.



## HV Communications

The HV Board has two RS485 interfaces. **Only one RS485 on the current ST365???**

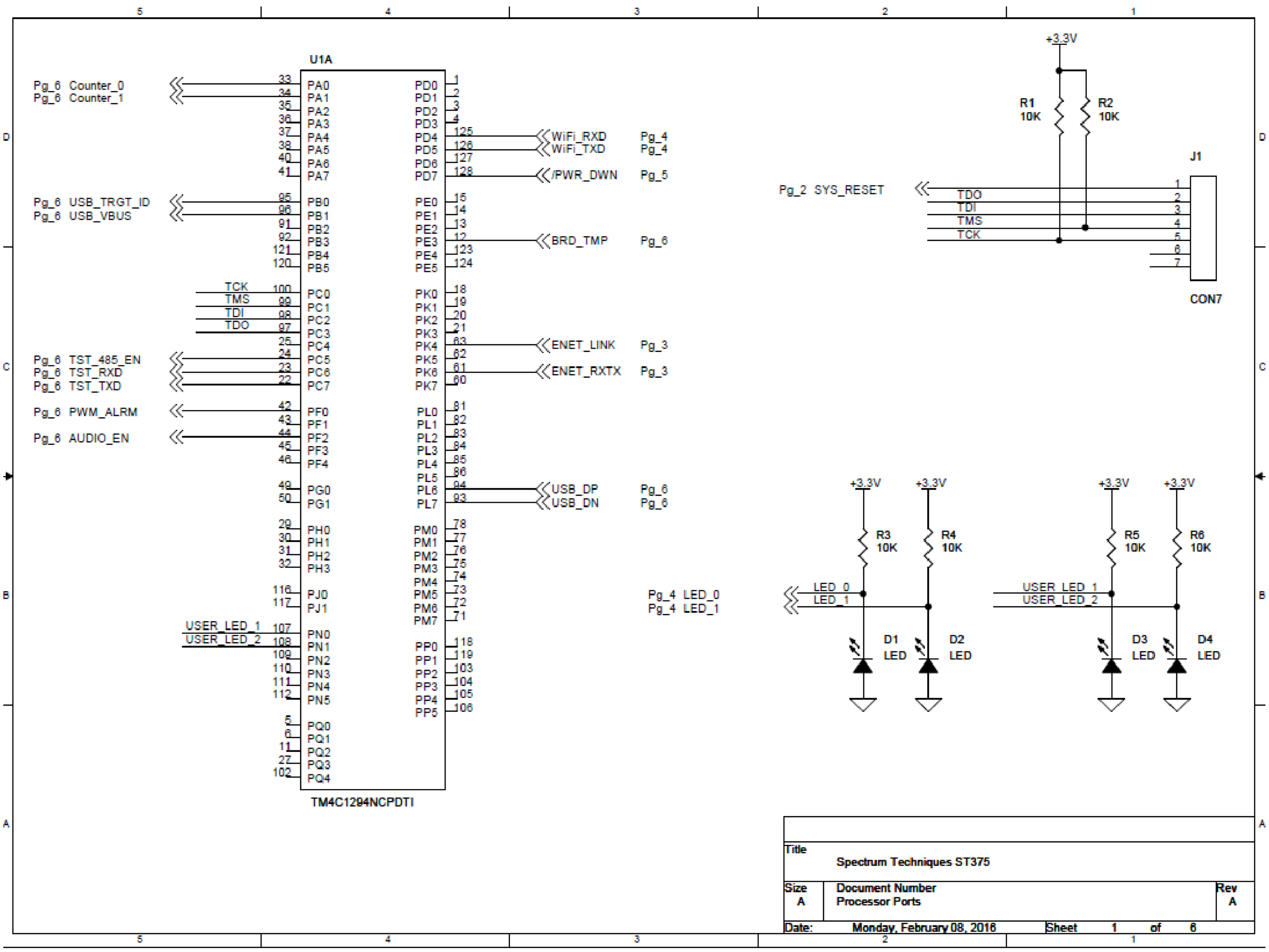


## Communications Board

This board acts as a bridge between the USB, Ethernet and WiFi communications interfaces and the STX. The Communications Board communicates with the SCA through a RS485 interface.  The Communications Board must arbitrate access to the RS485 port between the Ethernet, USB and WIFI interfaces. The SCA acts as the RS485 hub for the STX and distributes the messages to the Display and HV Boards.

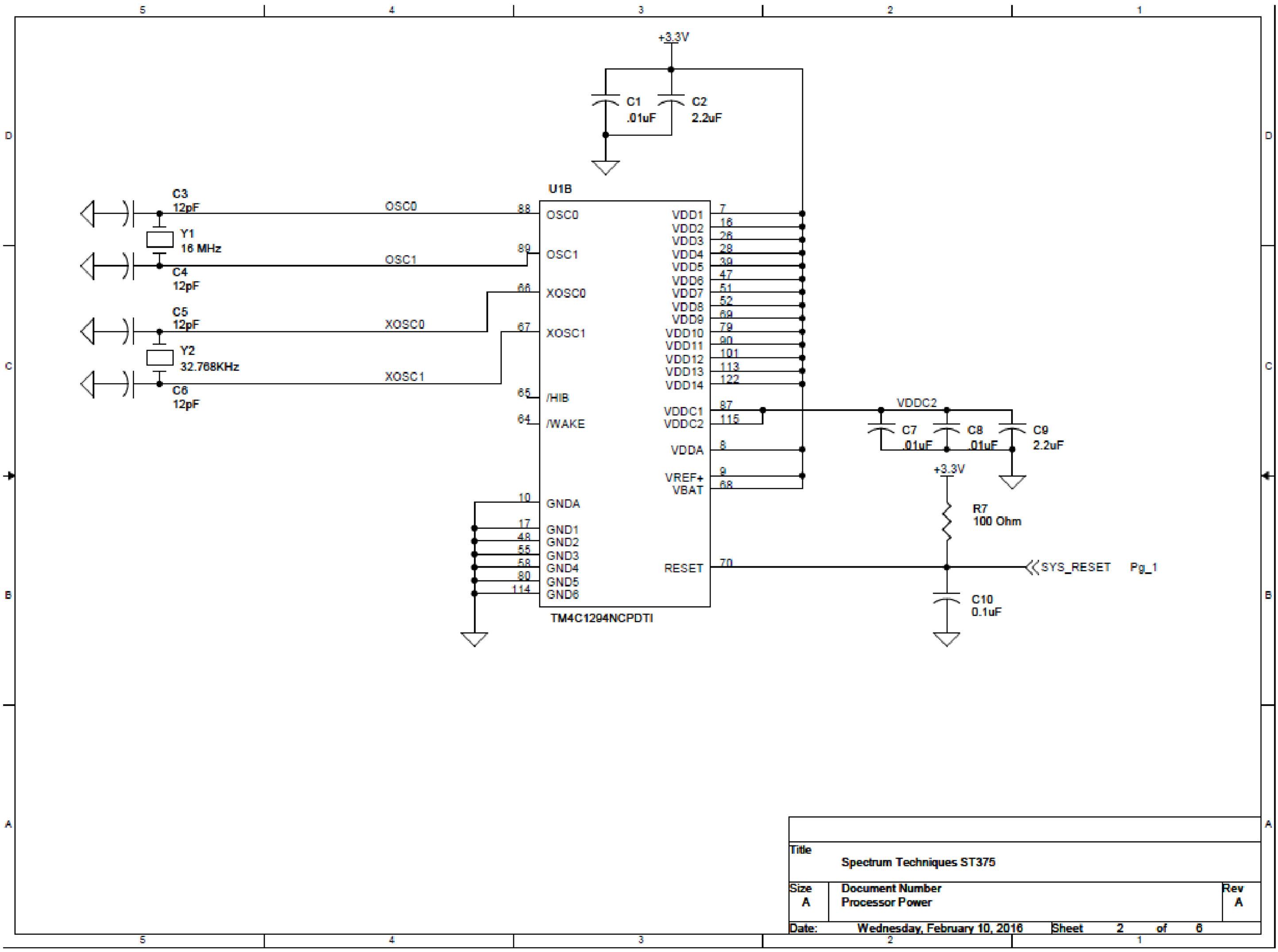
### Ethernet  Microcontroller Port I/O

The Ethernet controller is an **ARM® Cortex®-M4 Tiva processor** with a Ethernet 10/100Mbit interface. The Ethernet is on and listening, so the processor is exposed to all traffic on the network.



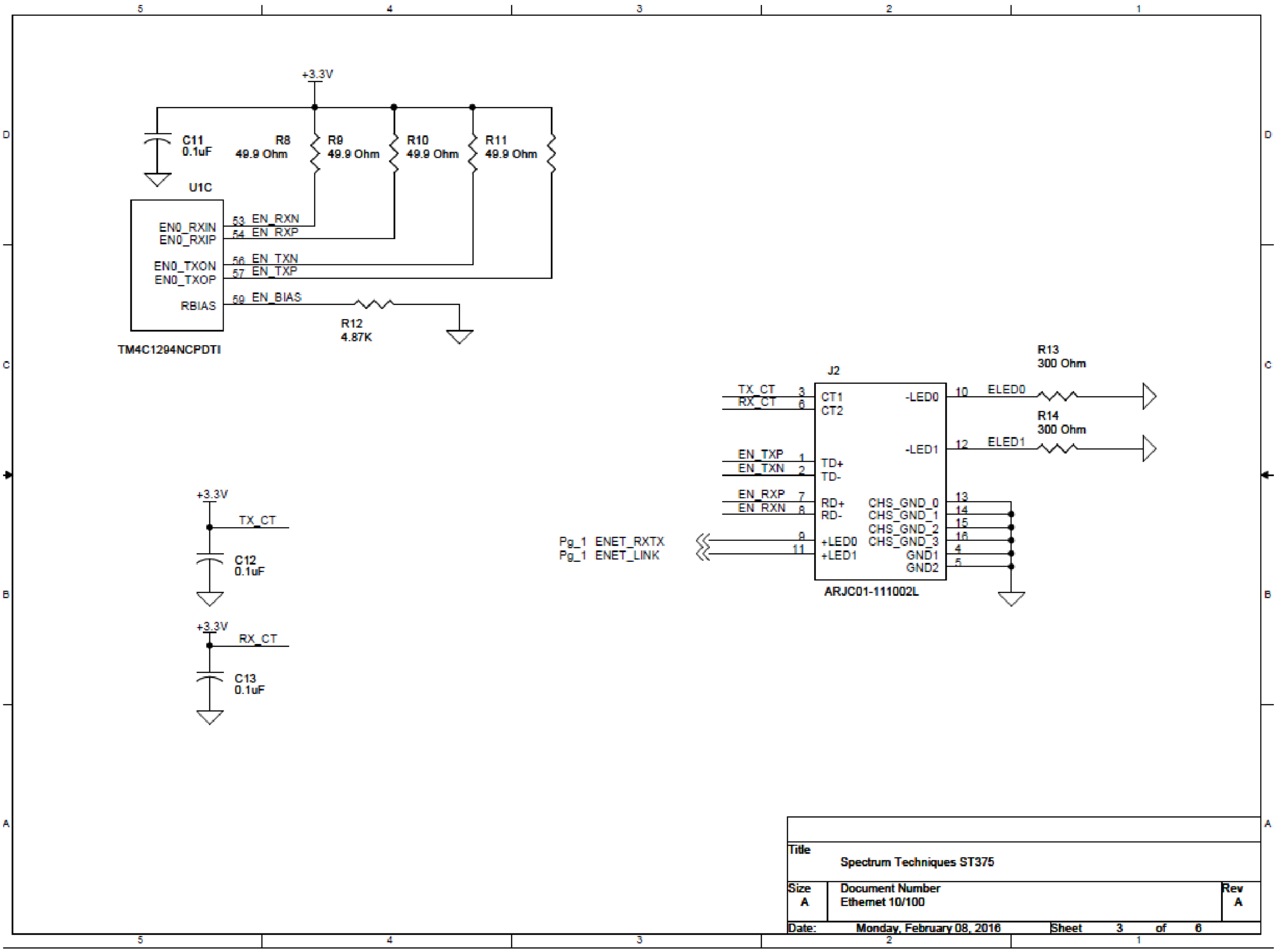
## Ethernet Power Dist and Crystals

This schematic is illustrates the power input pins and crystal oscillator balancing circuits.



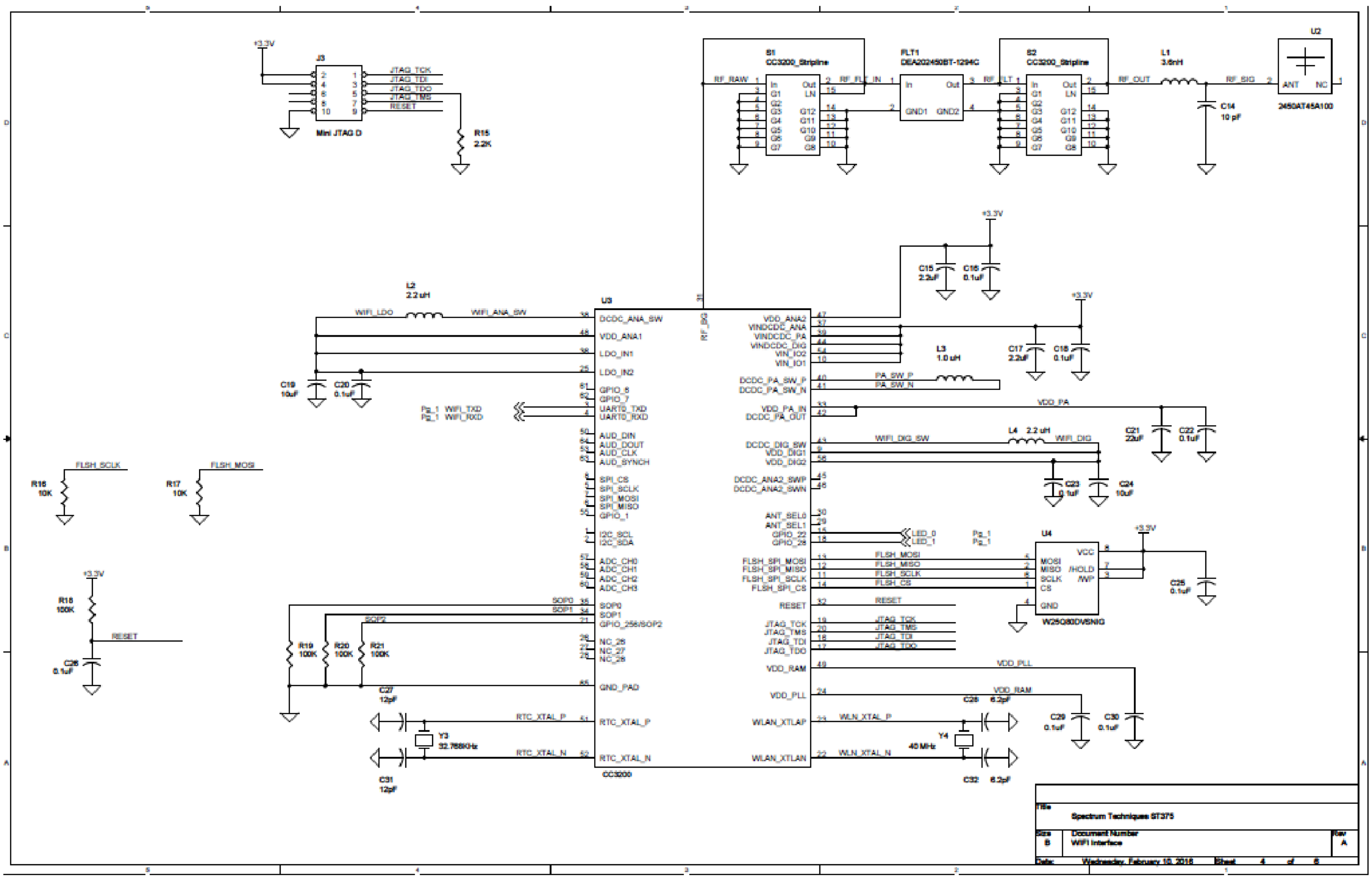
## Ethernet Port and Connector

This schematic is for the RJ45 10/100M Ethernet interface. This connector has the signal magnetics (transformers) for CAT5 or CAT6 cable built in.



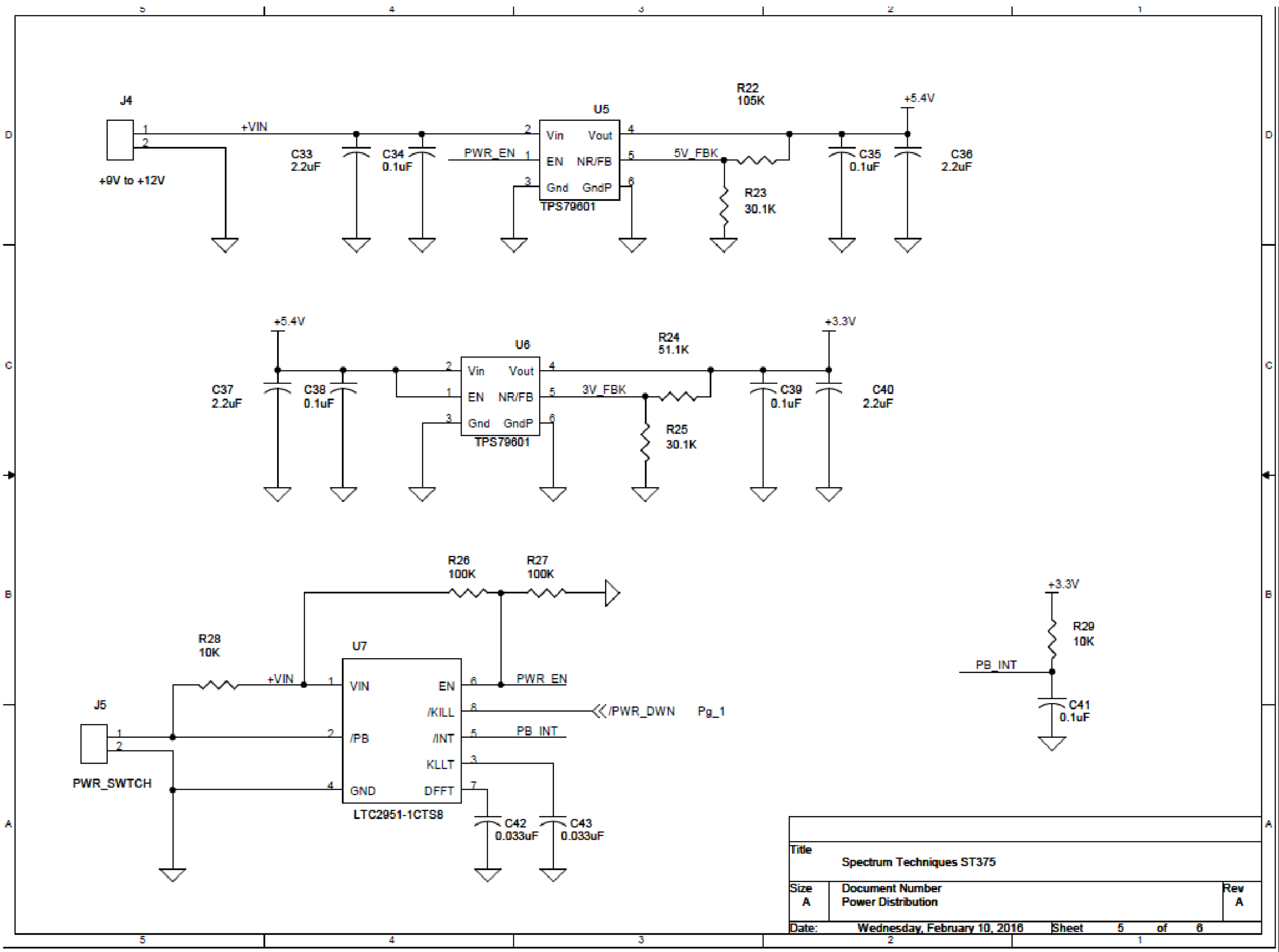
## WIFI co-processor and RF

This is the textbook implementation of the CC3200 WiFi interface from TI. The two devices labeled CC3200 Strip line are used to reference a footprint that has predetermined geometric dimensions and are set as devices to fool the CAD software into making them out of bounds for manipulation. This is so the geometries remain fixed, no actual component exists, but the resulting PCB footprint is a perfectly tuned strip line for the 2GHz RF interface. **2.4GHz???**



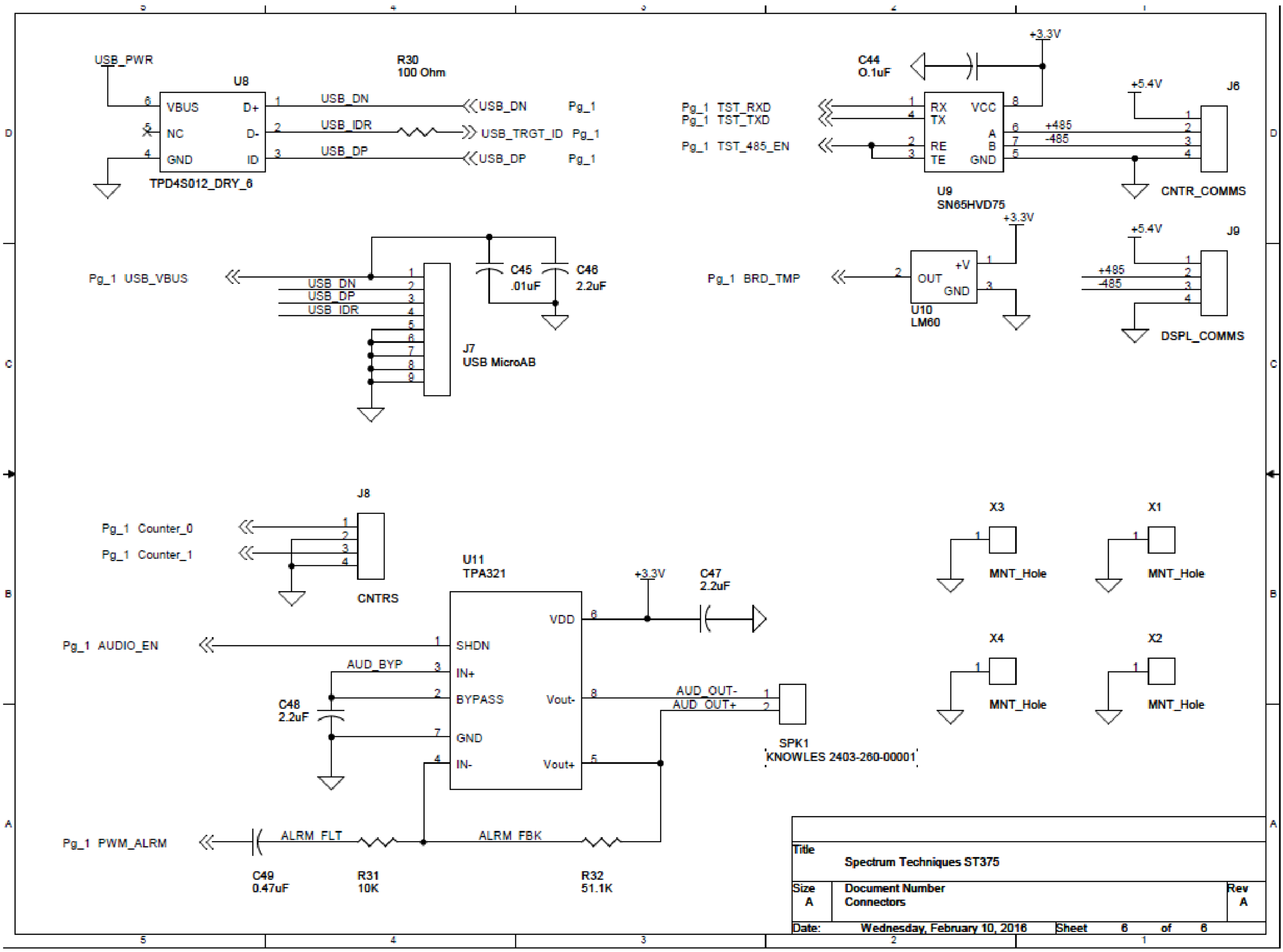
## Communications Board Power Regulators

This schematic of the power regulators for the Communications Board.



## Communications Board Connectors

This schematic holds the LEDS, connectors and RS485 interfaces. **Two RS485 ports??? USB Charger U8?? Audio circuit is no longer there.**



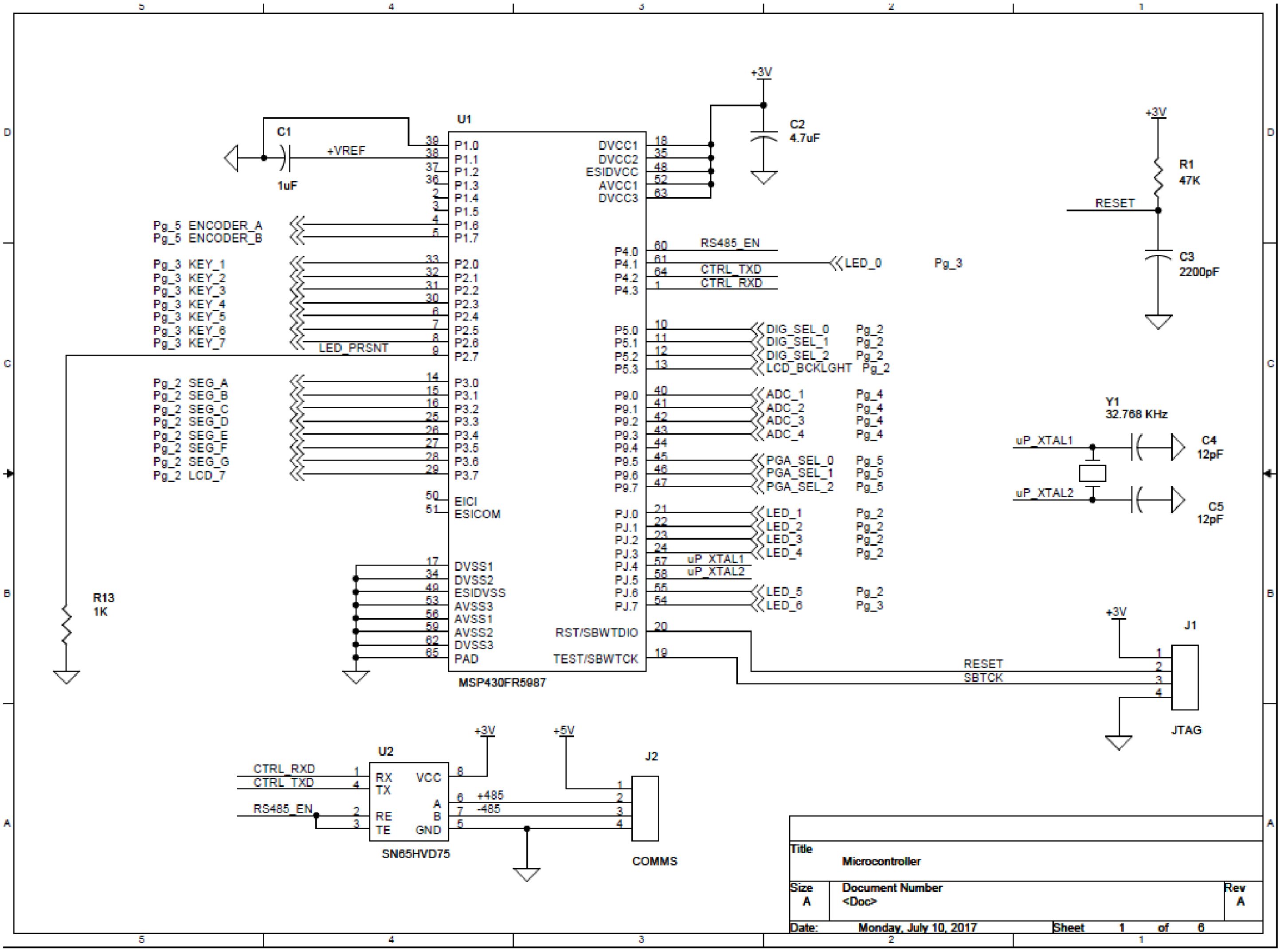
## Display Board

**The Display Board is dual use on two fronts:??????**

1) If the build includes the knobs and extra button the firmware loaded should be STX-475 version, **to include full SCA capabilities, widowing etc.**
2) LED can be inserted instead of the LCD, messages go away and the firmware should be re compiled to use the LED Mux library.

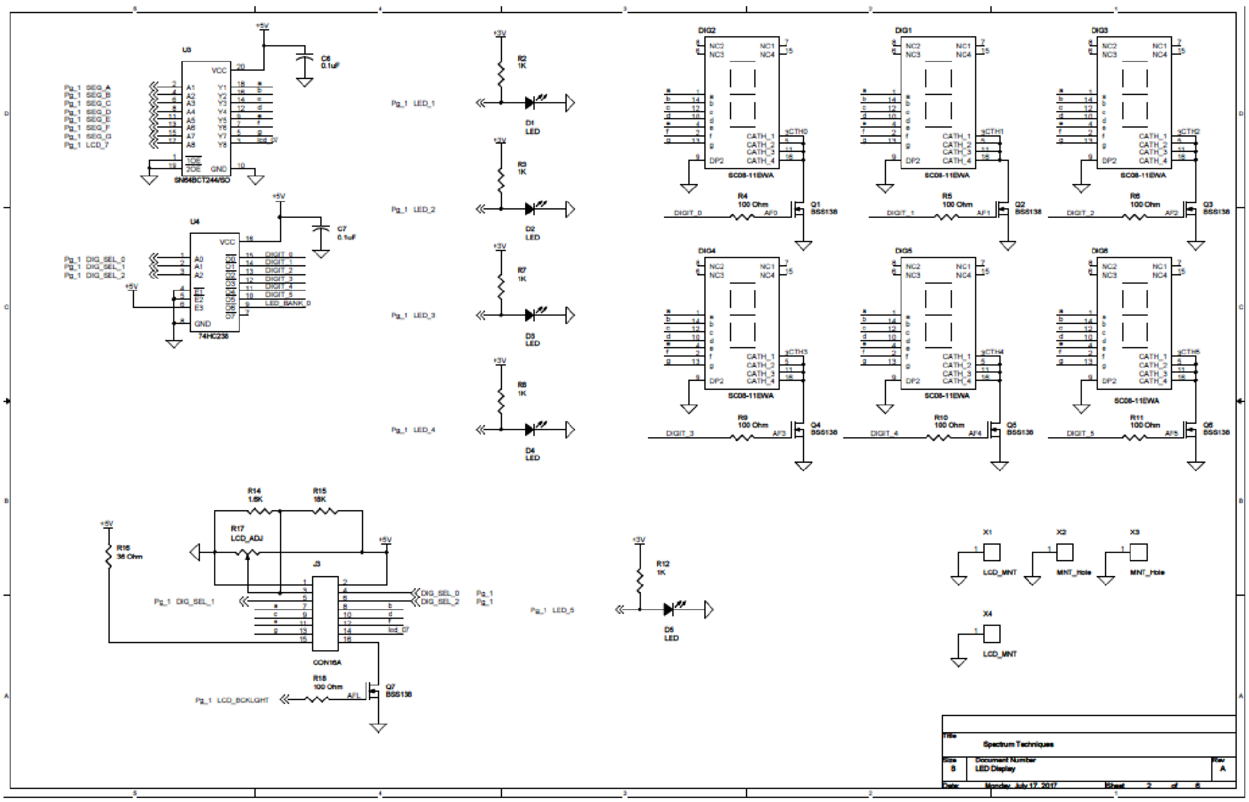
### Display Microcontroller

This is FRAM based so after bootup and initialization the screen and parameter settings will return to those used when power was turned off. HV will reset to Idle Off mode however, The remaining settings will be the same.


U1
MSP430FR5987
C1
1uF
+VREF
+3V
C2
4.7uF
R1
47K
RESET
C3
2200pF
Pg_5 ENCODER_A
Pg_5 ENCODER_B
Pg_3 KEY_1
Pg_3 KEY_2
Pg_3 KEY_3
Pg_3 KEY_4
Pg_3 KEY_5
Pg_3 KEY_6
Pg_3 KEY_7
LED_PRSNT
Pg_2 SEG_A
Pg_2 SEG_B
Pg_2 SEG_C
Pg_2 SEG_D
Pg_2 SEG_E
Pg_2 SEG_F
Pg_2 SEG_G
Pg_2 LCD_7
P1.0 P1.1 P1.2 P1.3 P1.4 P1.5 P1.6 P1.7
P2.0 P2.1 P2.2 P2.3 P2.4 P2.5 P2.6 P2.7
P3.0 P3.1 P3.2 P3.3 P3.4 P3.5 P3.6 P3.7
EICI
ESICOM
DVSS1 DVSS2 ESIDVSS AVSS3 AVSS1 AVSS2 DVSS3 PAD
DVCC1 DVCC2 ESIDVCC AVCC1 DVCC3
P4.0 P4.1 P4.2 P4.3
RS485_EN
CTRL_TXD
CTRL_RXD
LED_0 Pg_3
P5.0 P5.1 P5.2 P5.3
DIG_SEL_0 Pg_2
DIG_SEL_1 Pg_2
DIG_SEL_2 Pg_2
LCD_BCKLGHT Pg_2
P9.0 P9.1 P9.2 P9.3 P9.4 P9.5 P9.6 P9.7
ADC_1 Pg_4
ADC_2 Pg_4
ADC_3 Pg_4
ADC_4 Pg_4
PGA_SEL_0 Pg_5
PGA_SEL_1 Pg_5
PGA_SEL_2 Pg_5
PJ.0 PJ.1 PJ.2 PJ.3 PJ.4 PJ.5 PJ.6 PJ.7
LED_1 Pg_2
LED_2 Pg_2
LED_3 Pg_2
LED_4 Pg_2
uP_XTAL1
uP_XTAL2
LED_5 Pg_2
LED_6 Pg_3
RST/SBWTDIO
TEST/SBWTCK
Y1
32.768 KHz
C4
12pF
C5
12pF
R13
1K
J1
JTAG
RESET
SBTCK
U2
SN65HVD75
RX TX RE TE VCC A B GND
+5V
+485
-485
J2
COMMS
Title
Microcontroller
Size A
Document Number
<Doc>
Rev A
Date: Monday, July 10, 2017
Sheet 1 of 6

## Display 7-segment LEDs & interface

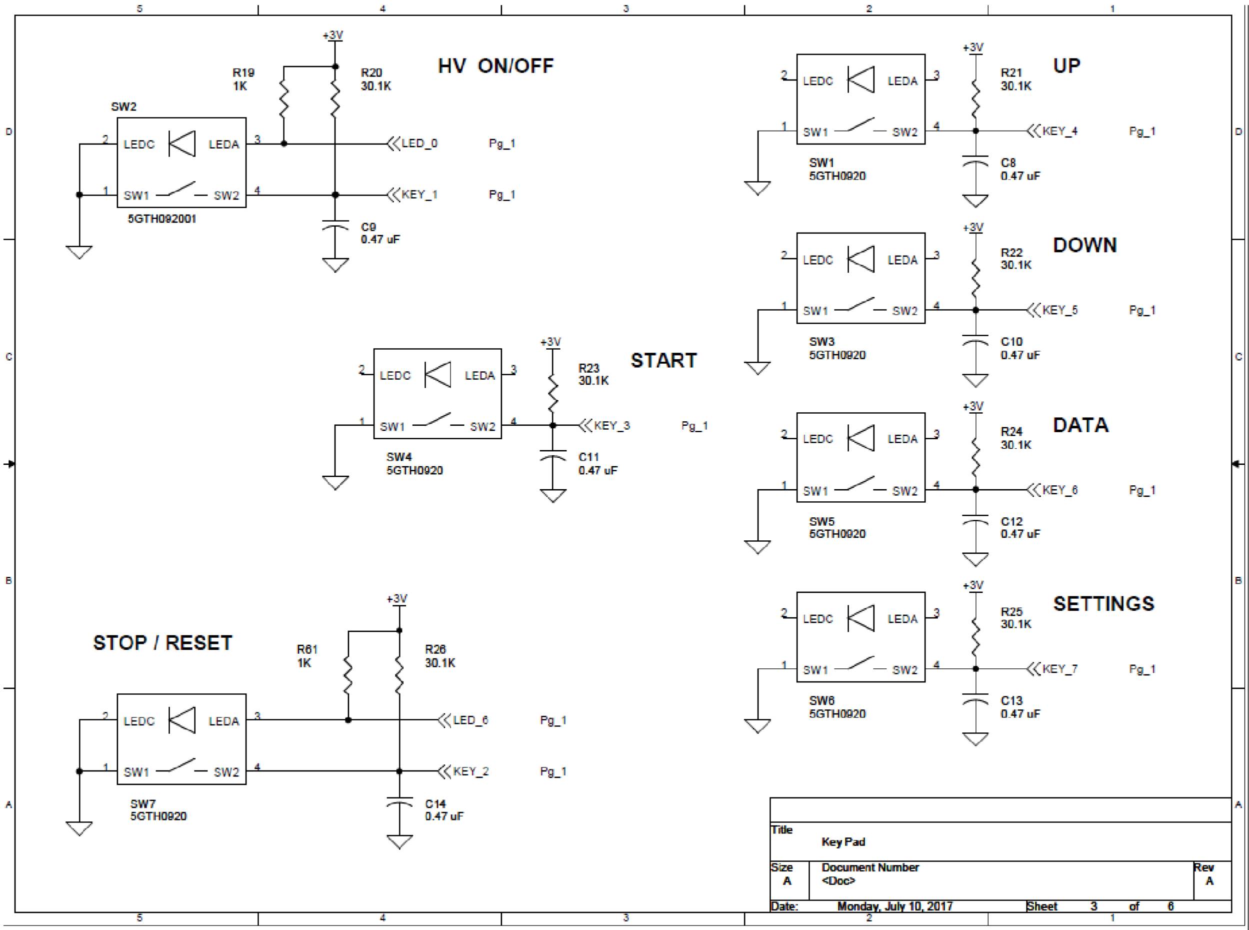
This schematic contains the LCD Level translator (3V to 5V) and the LED mux, common anode driver. R17 LCD contrast adj is legacy, only the fixed voltage divider is present in the production model.



## Display Buttons/Keys

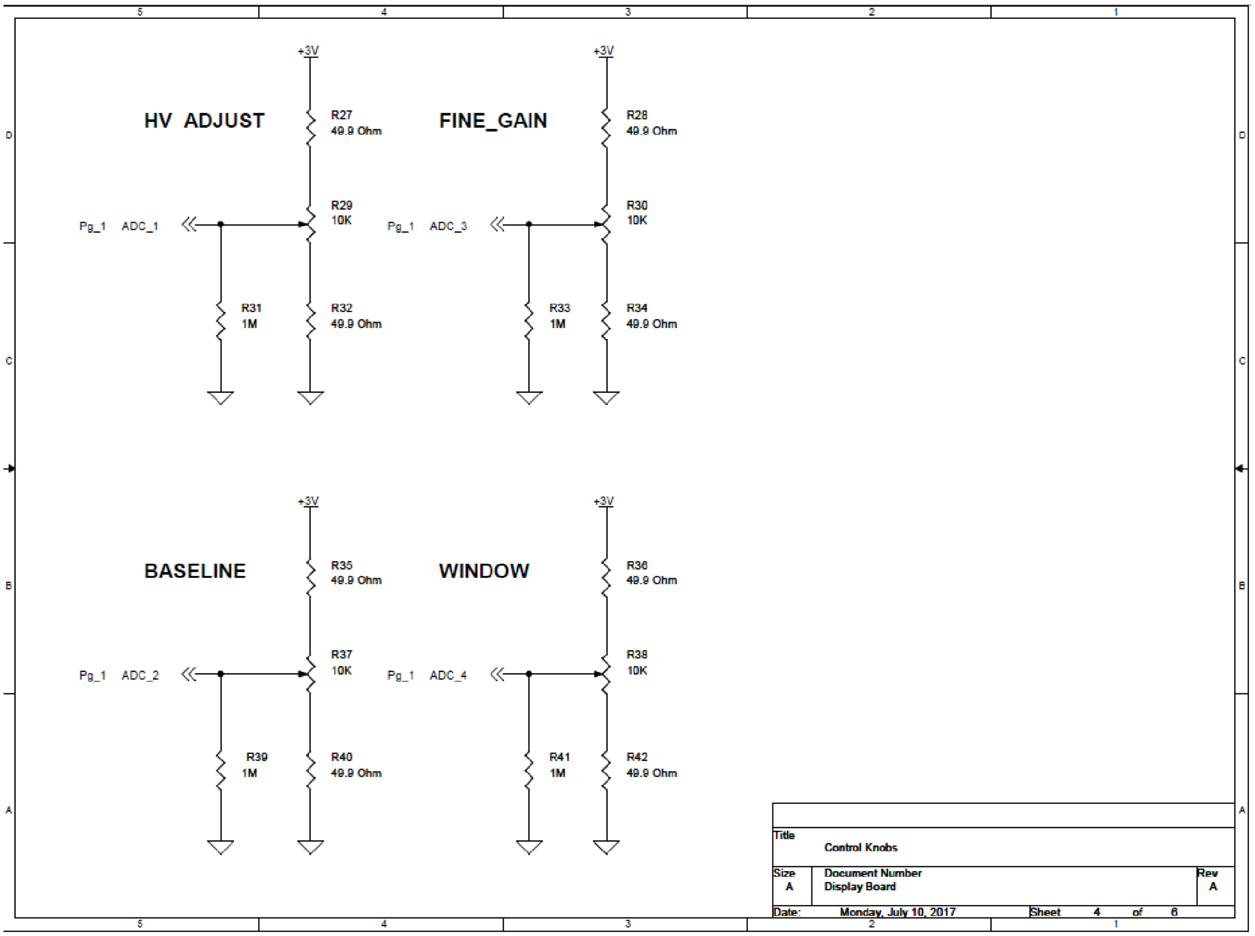
The Keys are interrupt driven non scanned. With exception of the power ON/OFF controller which resides on the always on side of the regulators.

The three rectangular keys have an embedded LED, the five navigation keys have no LED.



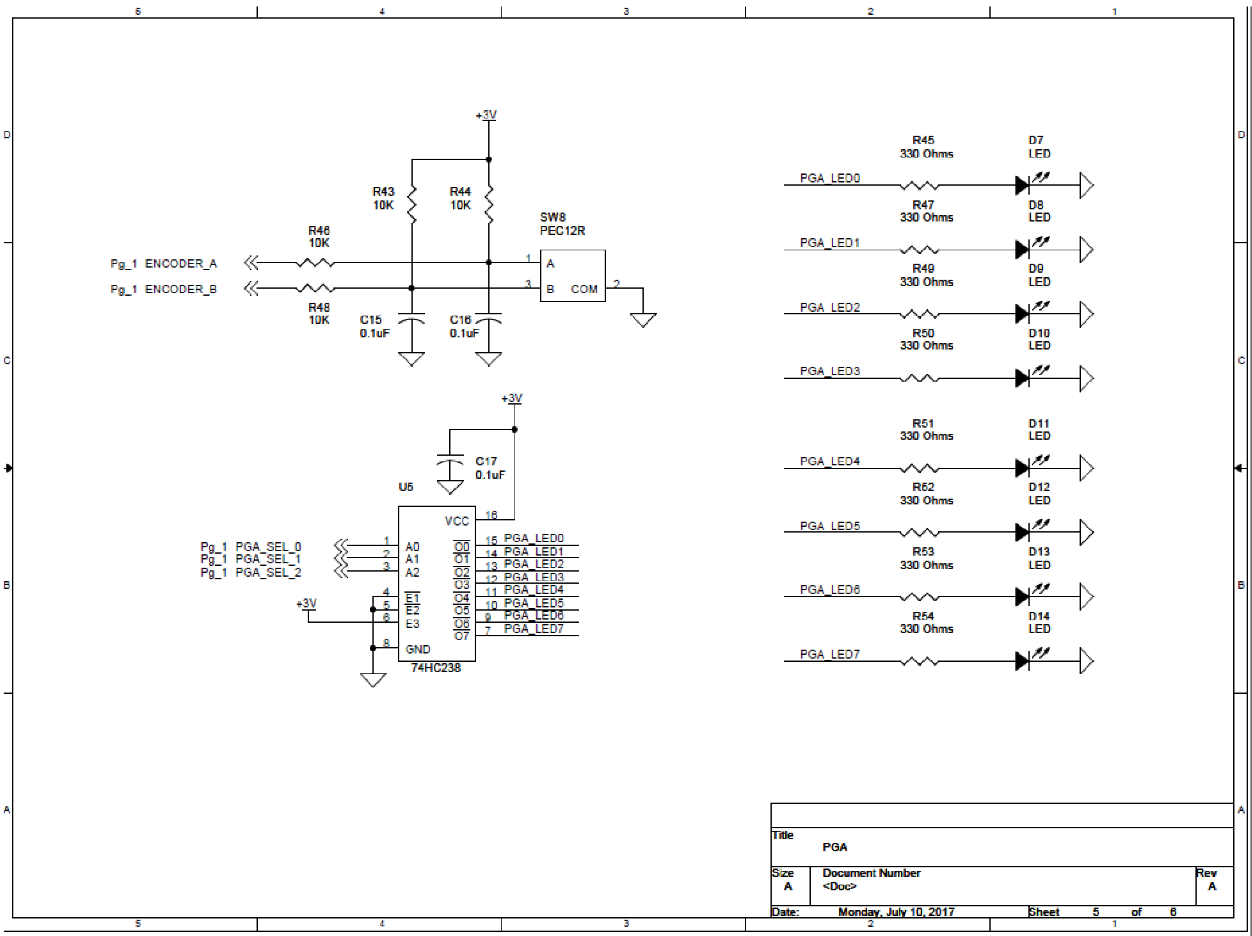
## Display Knobs

All of the Knobs are read using the microcontroller ADC, the 1 M resistor should be replaced with a 4.7uF cap to de-bounce. Internally a 5 sample moving average filter is used to smooth the data. The ADC is 12 Bits but the range is limited and the resulting calculation is smoothed to the closest approximation of the desired setting. **On the STX-365 these knobs are not populated. ST365??**



## Display PGA Settings

The user interface for the PGA can either be menu driven directly on the LCD or provided with a two bit encoder (knob) that increments and decrements the gain selection. In this mode the corresponding LED indicates the gain setting sent to the SCA module for use by the operational amplifiers that process the pulses from the radiation detectors.



## PGA Display Settings Power

This schematic contains power regulators, and the ON/Off main power controller.