University of Nottingham
UK | CHINA | MALAYSIA

University of Nottingham FYP Progress Review

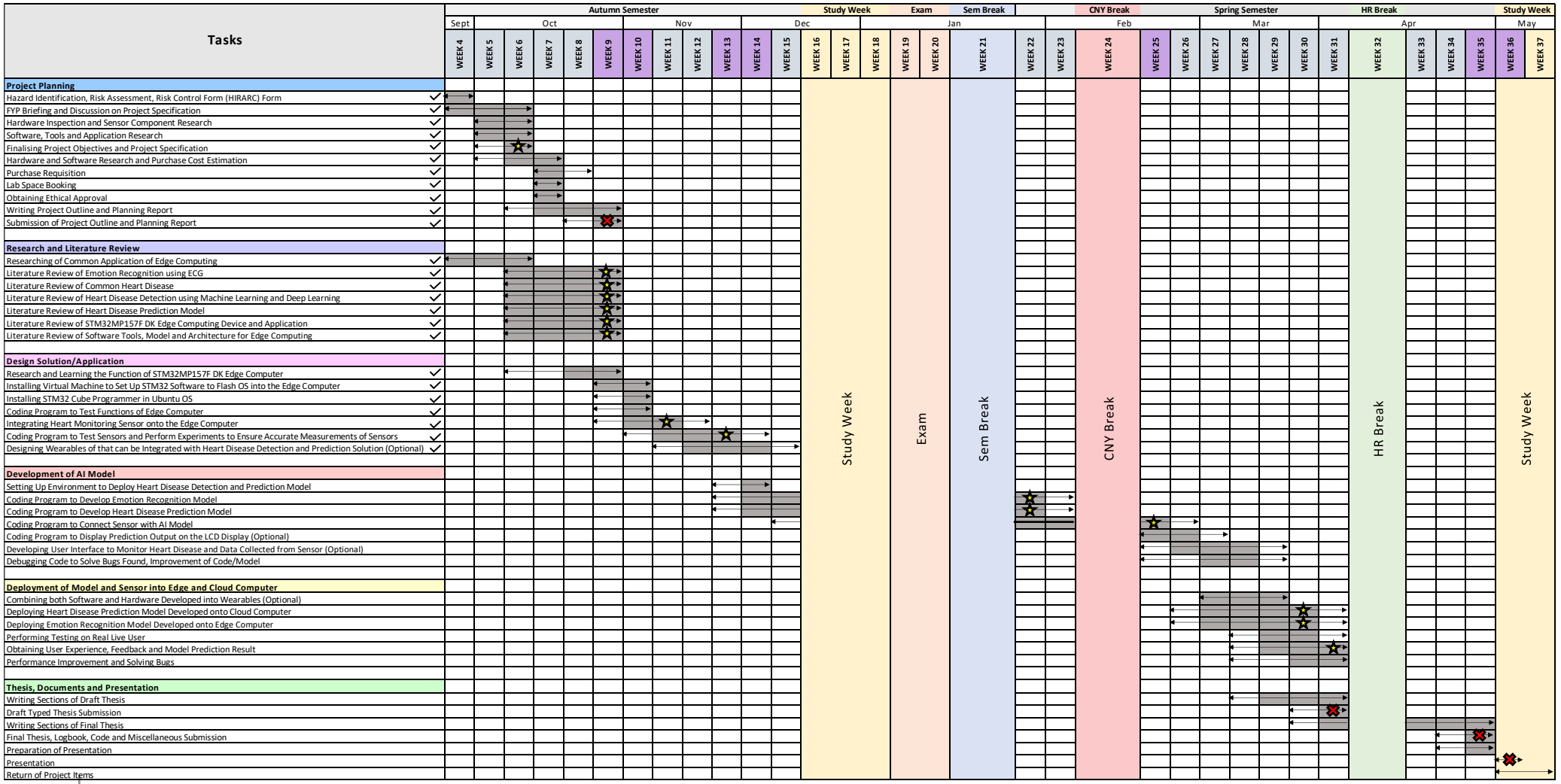
December 2023

Gantt Chart

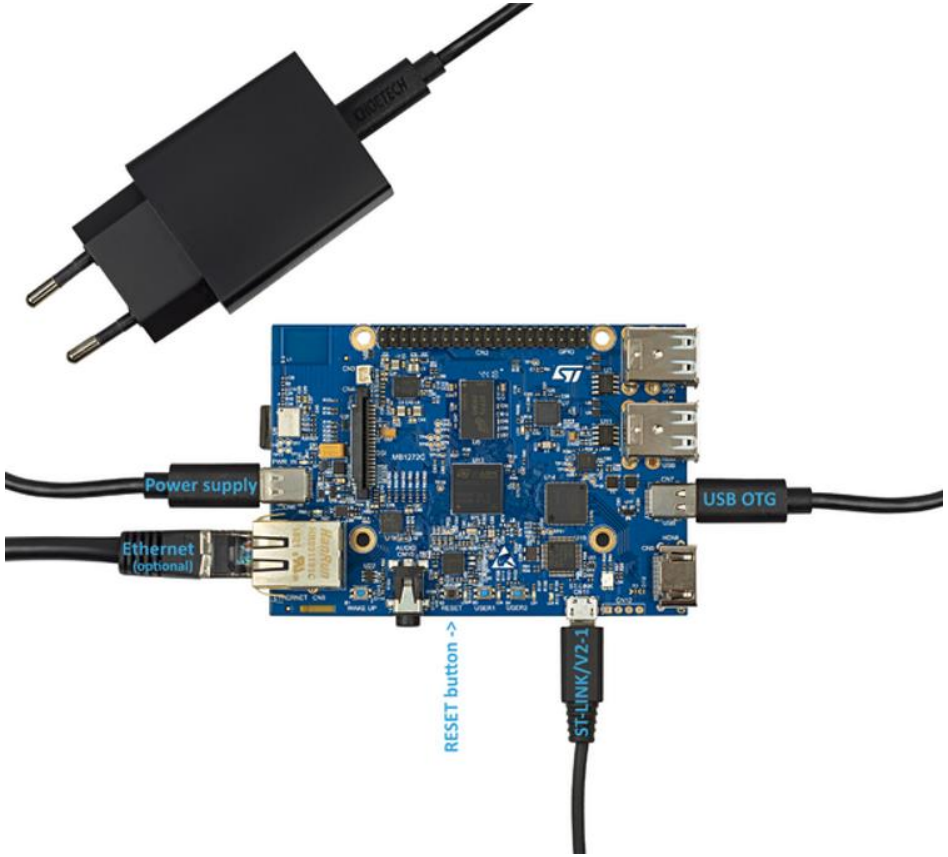
Supervisor: Dr Hermawan Nugroho
Moderator: Prof T. Nandha Kumar
Project Code: HN-BEng-23-01

Student Name: Koay Xian Cong
Student ID: 20418760

Legends: Error Bars for Uncertainty: ↔ Moderator Meeting Week:
Submission Dateline: ✖ Project Milestone: ★

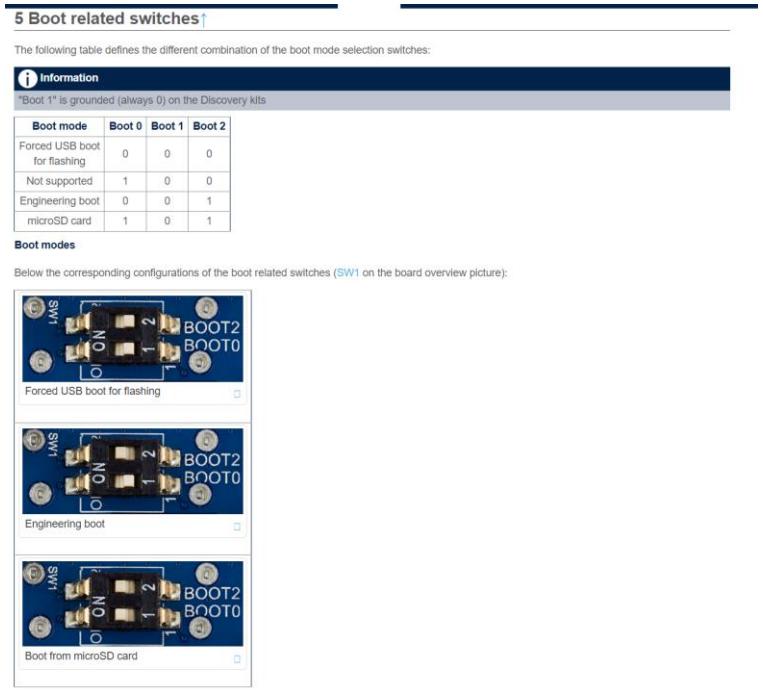


Setting Up STM32CubeIDE & Testing Code



Requirements

- 5V 3A power supply is needed
- ST-Link used as a debugger connection
- Engineering Mode/Boot Mode



5 Boot related switches

The following table defines the different combination of the boot mode selection switches:

Information

"Boot 1" is grounded (always 0) on the Discovery kits

Boot mode	Boot 0	Boot 1	Boot 2
Forced USB boot for flashing	0	0	0
Not supported	1	0	0
Engineering boot	0	0	1
microSD card	1	0	1

Boot modes

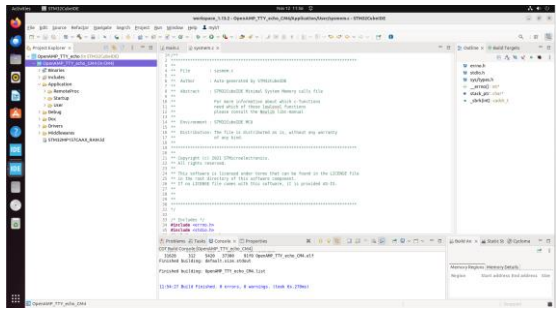
Below the corresponding configurations of the boot related switches (SW1 on the board overview picture):

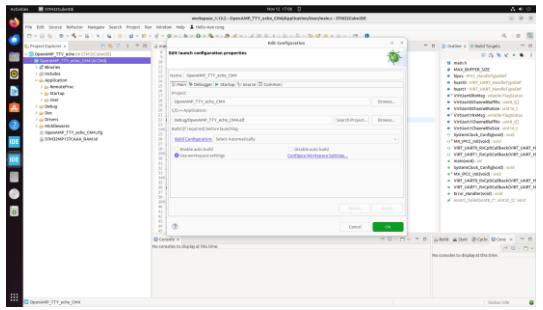
Forced USB boot for flashing

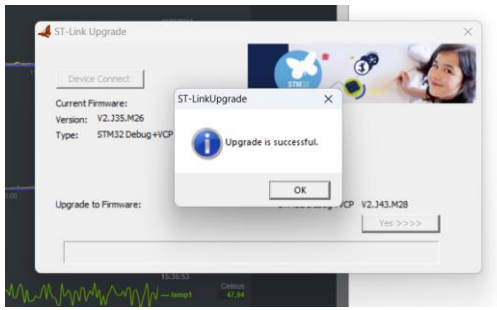
Engineering boot

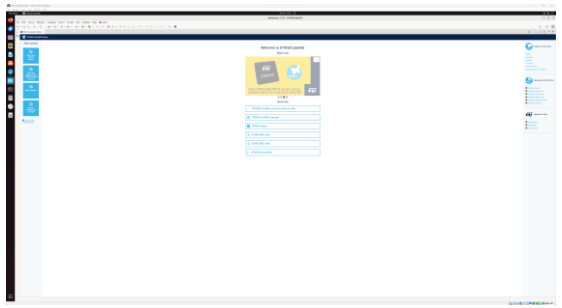
Boot from microSD card

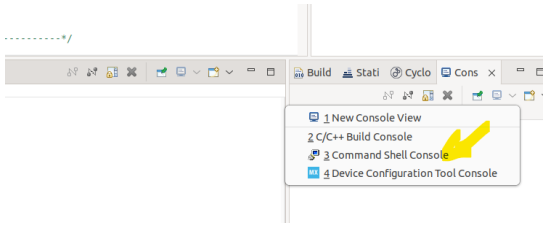
Setting Up STM32CubeIDE & Testing Code

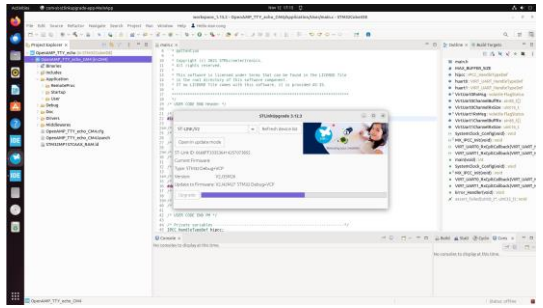


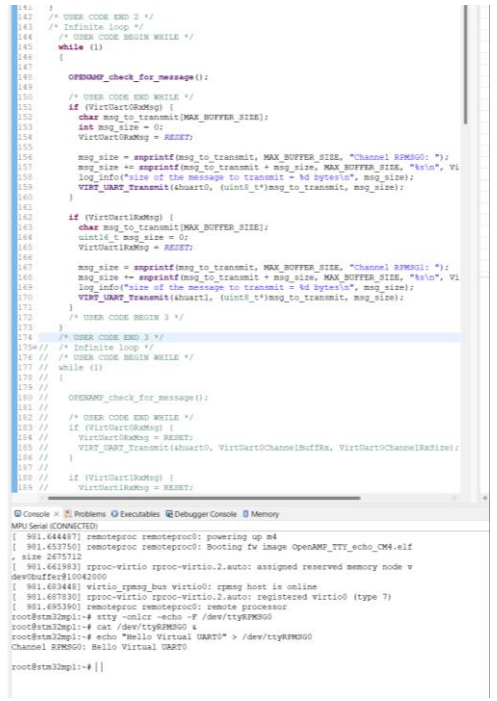


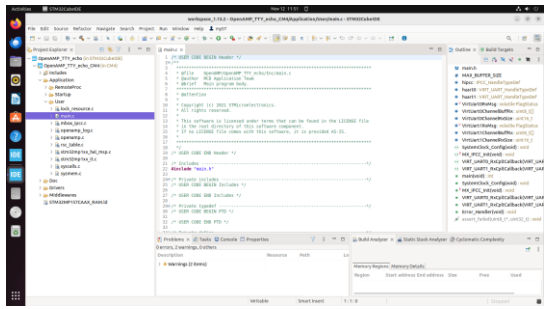


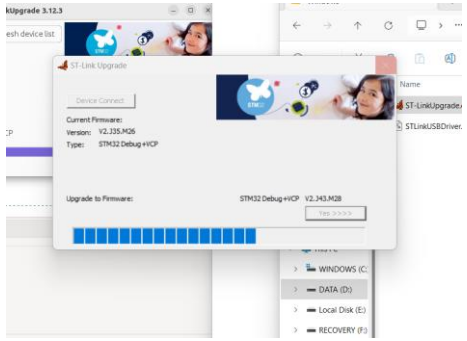






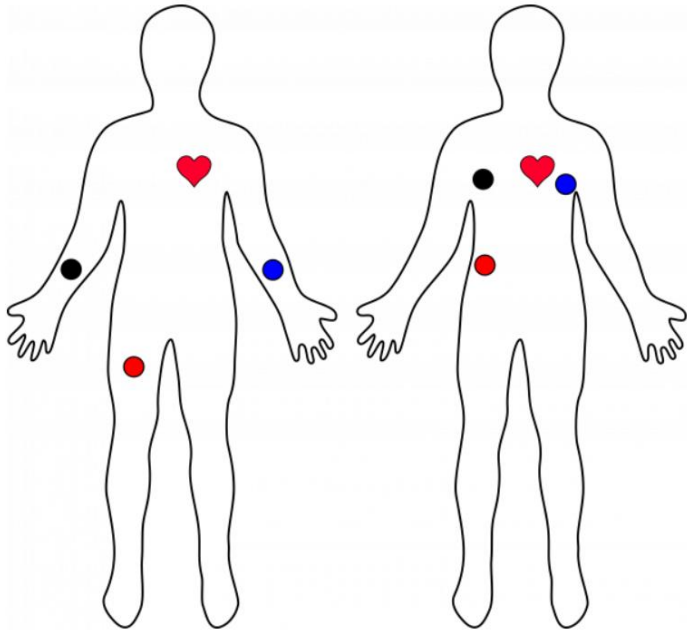




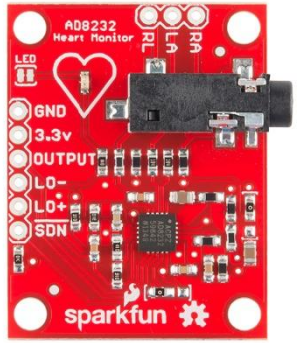


Connecting ECG sensors onto STM32MP157F-DK2

Cable Color	Signal
Black	RA (Right Arm)
Blue	LA (Left Arm)
Red	RL (Right Leg)



Typical Sensor Placements



Board Label	Pin Function	Arduino Connection
GND	Ground	**GND**
3.3v	3.3v Power Supply	**3.3v**
OUTPUT	Output Signal	**A0**
LO-	Leads-off Detect -	**11**
LO+	Leads-off Detect +	**10**
SDN	Shutdown	**Not used**

```
void setup() {
Serial.begin(9600);
  pinMode(10, INPUT); // Setup for leads off detection LO +
  pinMode(11, INPUT); // Setup for leads off detection LO -
}

void loop() {
  if((digitalRead(10) == 1)||(digitalRead(11) == 1)){
    Serial.println('!');
  }
  else{
    // send the value of analog input 0:
      Serial.println(analogRead(A0));
  }
  //Wait for a bit to keep serial data from saturating
  delay(1);
}
```

Connecting ECG sensors onto STM32MP157F-DK2

6.16.3 ARDUINO® interface

Figure 16 shows the pinout of the ARDUINO® connectors.

Figure 16. **ARDUINO® connectors pinout**

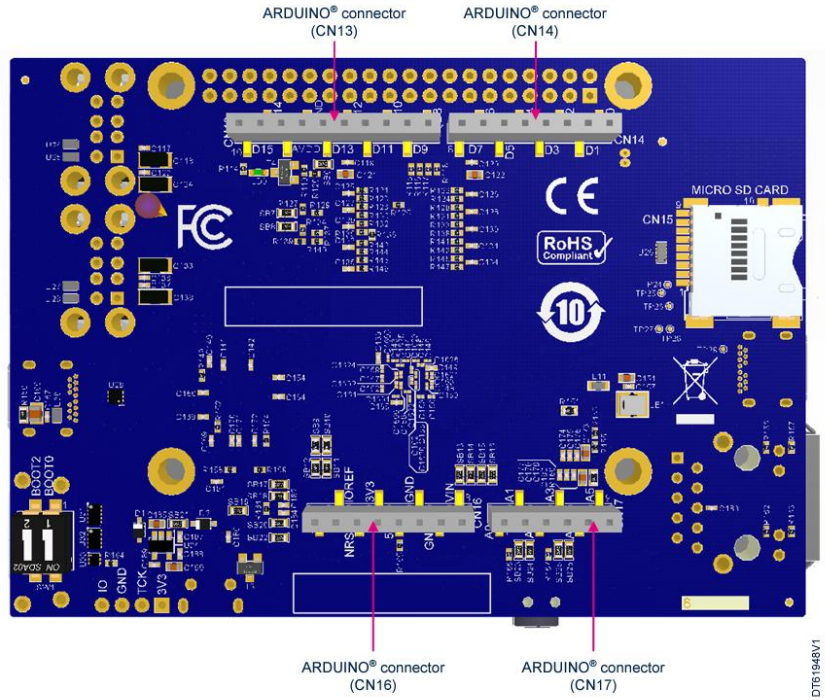


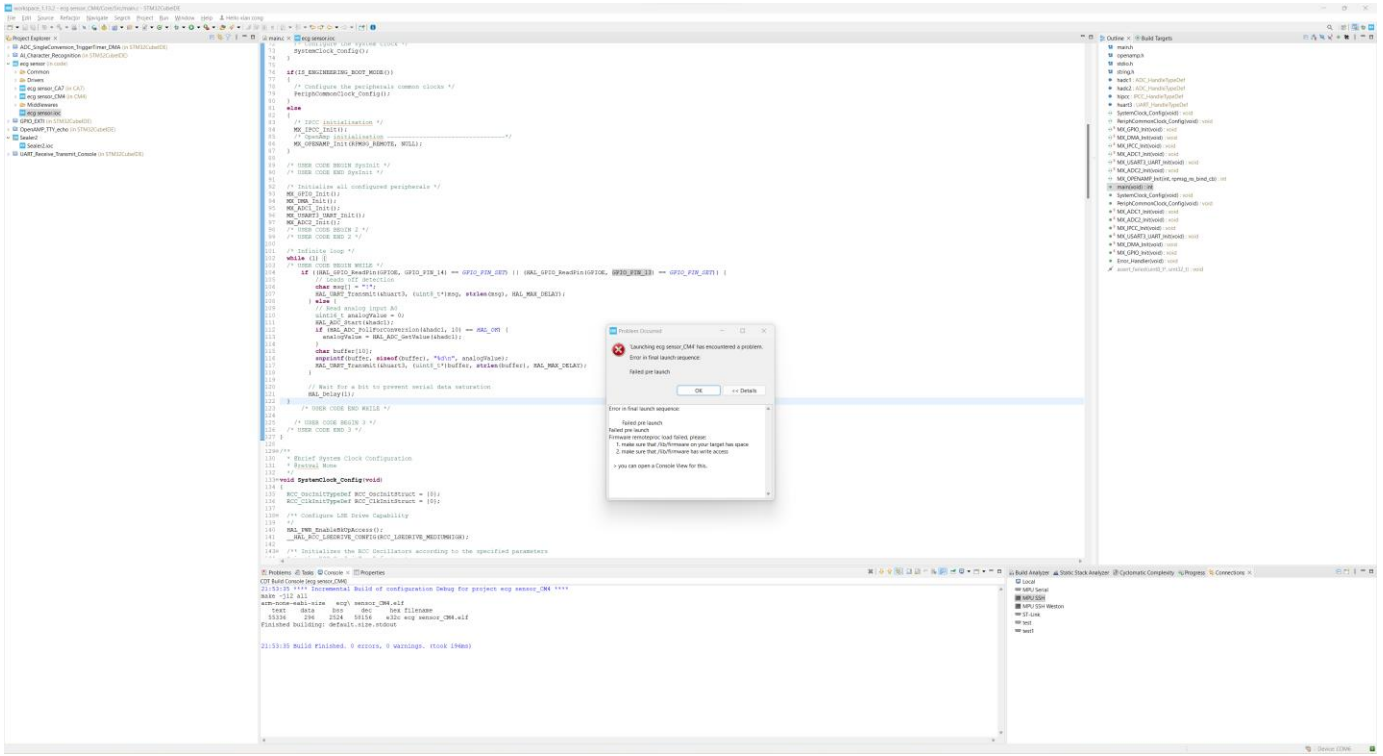
Table 23 describes the I/O configuration of the ARDUINO® interface.

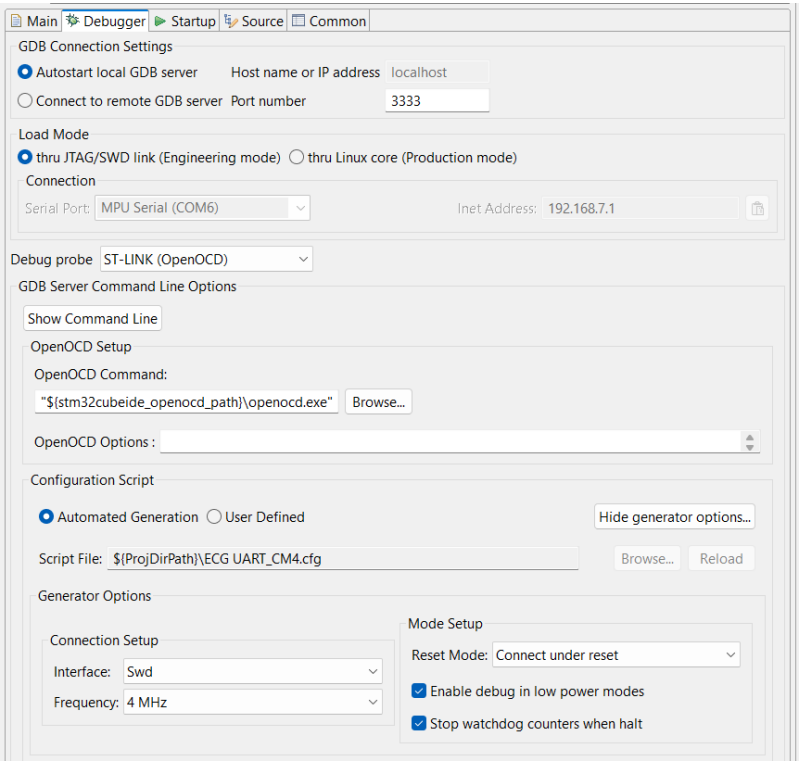
Table 23. **I/O configuration of the ARDUINO® interface**

I/O	HW	Configuration
PF14	-	PF14 is used as ARD_A0: ADC2_IN6
PF13	-	PF13 is used as ARD_A1: ADC2_IN2
ANA0	-	ANA0 is used as ARD_A2: ADC1_IN0, ADC2_IN0
ANA1	-	ANA1 is used as ARD_A3: ADC1_IN1, ADC2_IN1
PC3	SB24 ON	PC3 is used as ARD_A4: ADC1_IN13 default configuration
PF12	SB26 ON	PF12 is used as ARD_A5: ADC1_IN16 default configuration
PE7	-	PE7 is used as ARD_D0: UART7_RX
PE8	-	PE8 is used as ARD_D1: UART7_TX

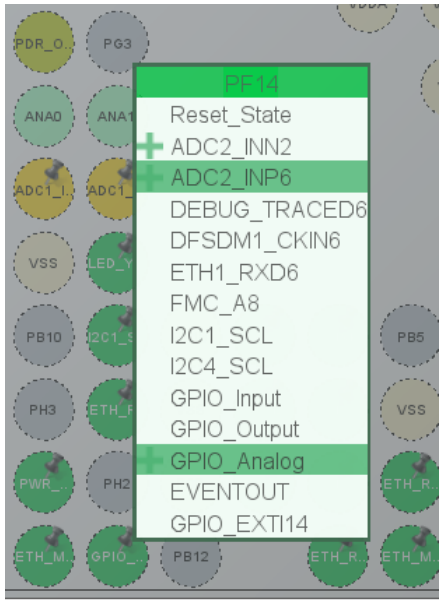
Connector	Pin name	Signal name	STM32 pin	Comment
CN13	1	ARD_D8	PG3	IO
	2	ARD_D9	PH6	TIM12_CH1
	3	ARD_D10	PE11	SPI4_NSS and TIM1_CH2
	4	ARD_D11	PE14	SPI4_MOSI and TIM1_CH4
	5	ARD_D12	PE13	SPI4_MISO
	6	ARD_D13	PE12	SPI4_SCK
	7	GND	-	GND
	8	VREFP	-	VREF+
	9	ARD_D14	PA12	I2C5_SDA
	10	ARD_D15	PA11	I2C5_SCL

Problems

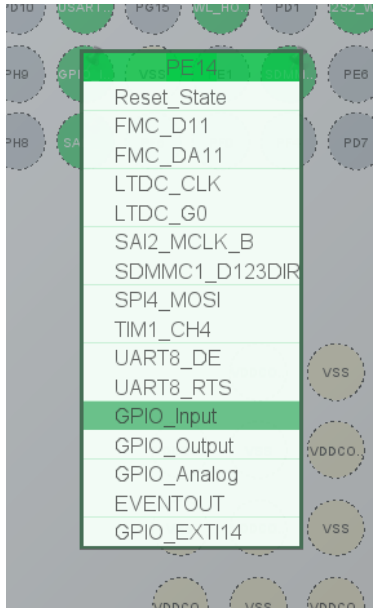



Main
Debugger
Startup
Source
Common
GDB Connection Settings
Autostart local GDB server
Host name or IP address
localhost
Connect to remote GDB server
Port number
3333
Load Mode
thru JTAG/SWD link (Engineering mode)
thru Linux core (Production mode)
Connection
Serial Port:
MPU Serial (COM6)
Inet Address:
192.168.7.1
Debug probe
ST-LINK (OpenOCD)
GDB Server Command Line Options
Show Command Line
OpenOCD Setup
OpenOCD Command:
"${stm32cubeide_openocd_path}\openocd.exe"
Browse...
OpenOCD Options :
Configuration Script
Automated Generation
User Defined
Hide generator options...
Script File:
${ProjDirPath}\ECG UART_CM4.cfg
Browse...
Reload
Generator Options
Connection Setup
Interface:
Swd
Frequency:
4 MHz
Mode Setup
Reset Mode:
Connect under reset
Enable debug in low power modes
Stop watchdog counters when halt

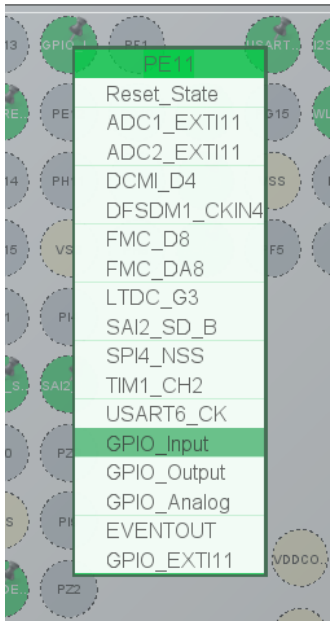
STM32 Pin Configuration



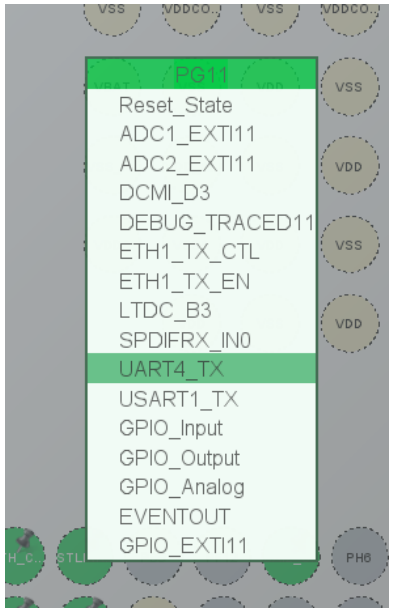
ADC

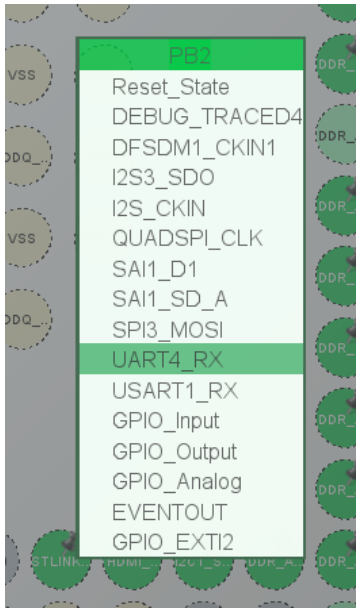


LO-



LO+





ST-Link UART

ADC

6 REPLIES

Sort: Oldest to Newest

PatrickF
ST Employee

2022-09-28 06:43 AM

Hi @Andrés Bonilla

Except if you have very tight real-time constrains, you should probably being able to do what you want by using STM32MP157F-DK2 starter package and Cortex-A7 Linux.

Cortex-M4 is usually required when there is hard real time constrains (e.g. motor control, high input reactivity, etc...).

Some ADC and TTY(UART) trials with Linux command lines are possible

(note that you need to choose '2: stm32mp157f-dk2-a7-examples' during boot to get confirmation listed below) :

see https://wiki.st.com/stm32mpu/wiki/How_to_use_the_IIO_user_space_interface#How_to_do_a_simple_ADC_conversion_using_the_sysfs_interface

Some ADC pins are available with this starter package Device Tree on DK2 board (please also refer to board documentation

- ADC1_INP0 and ADC2_INP0 on pin ANA0 (Arduino connector A2)
- ADC1_INP1 and ADC2_INP0 on pin ANA1 (Arduino connector A3)
- ADC1_INP6 on pin PF12 (Arduino connector A5)
- ADC1_INP13 on pin PC3 (Arduino connector A4)
- ADC2_INP2 on pin PF13 (Arduino connector A1)
- ADC2_INP6 on PF14 (Arduino connector A0)

For UART,

See https://wiki.st.com/stm32mpu/wiki/Serial_TTY_overview

- ttySTM1 is USART3 (RX/TX/RTS/CTS on GPIO connector)
- ttySTM2 is UART7 (RX/TX on Arduino connector)

Then, you could use them from any application (e.g. in python).

For GUI, I'm not expert, but there is many possible options with Weston Wayland (https://wiki.st.com/stm32mpu/wiki/Wayland_Weston_overview).

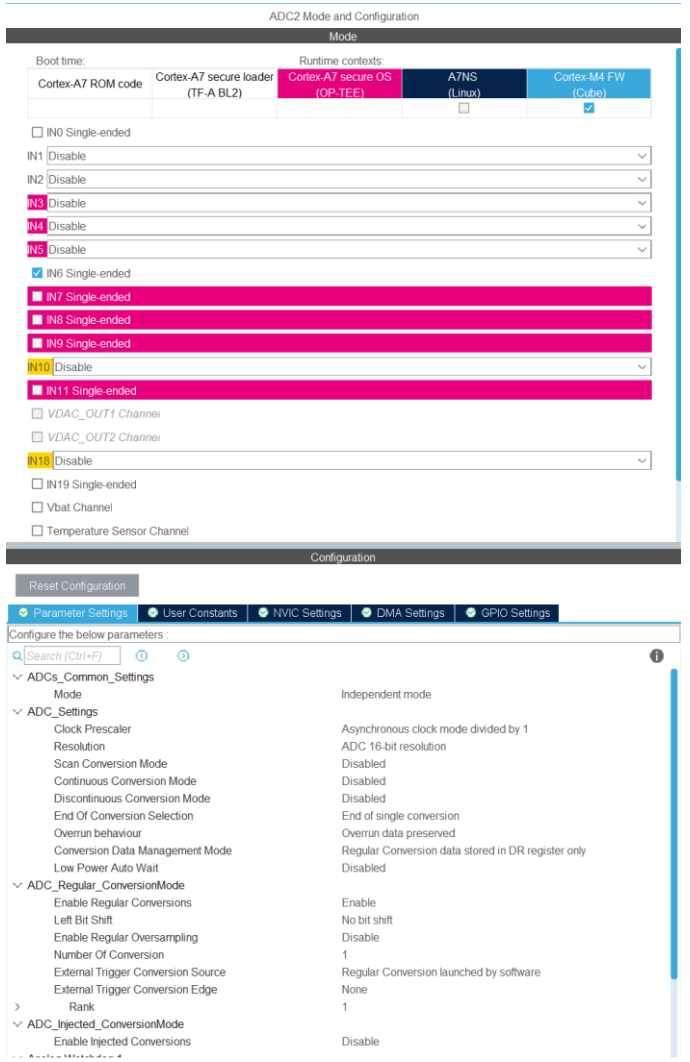
Demo is using GTK (https://wiki.st.com/stm32mpu/wiki/GTK_demo_launcher), but there is certainly other options.

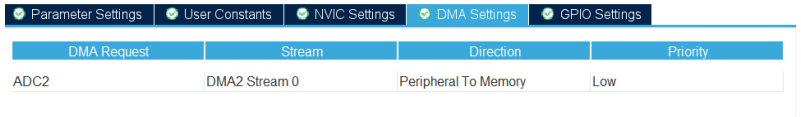
Regards

In order to give better visibility on the answered topics, please click on ***'Accept as Solution'*** on the reply which solved your issue or answered your question.

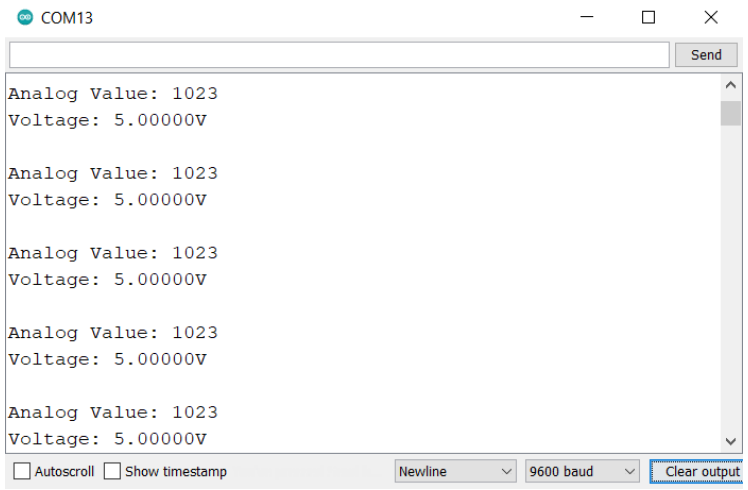
0 Kudos

Reply





Problem



Kevin HUBER
ST Employee

2022-02-10 07:17 AM

Hi @msche.1 ,

Thanks to have shared your .ioc.

To be able to get a usable value by reading the ADC, you must have a reference which is VREFBUF.

I advice you to read the reference manual (RM0436 revision 5) page 1693:
https://www.st.com/resource/en/reference_manual/DM00327659-.pdf

and the ADC page on the wiki: https://wiki.st.com/stm32mpu/wiki/ADC_internal_peripheral#Features

Then you have two choices to enable VREFBUF.

- Engineering mode (without Linux)

If you are not using linux you have to enable the VREFBUF after the ADC_Init() but before starting it:

```
  __HAL_RCC_VREF_CLK_ENABLE(); // Enable the VREF clock
  HAL_SYSCFG_VREFBUF_HighImpedanceConfig(SYSCFG_VREFBUF_HIGH_IMPEDANCE_DISABLE); // Disable the high impedance
mode which is the default one read page 1694 of refman
  HAL_SYSCFG_VREFBUF_VoltageScalingConfig(SYSCFG_VREFBUF_VOLTAGE_SCALE1); // To set the volage to 2.5v
  HAL_SYSCFG_EnableVREFBUF(); // To enable VREFBUF
```

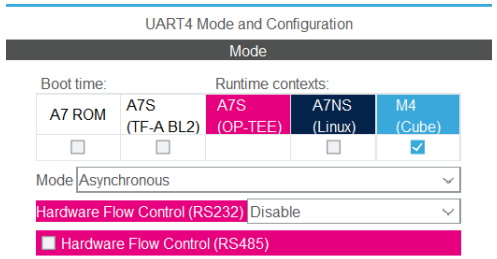
Then you must calibrate your adc before using it:

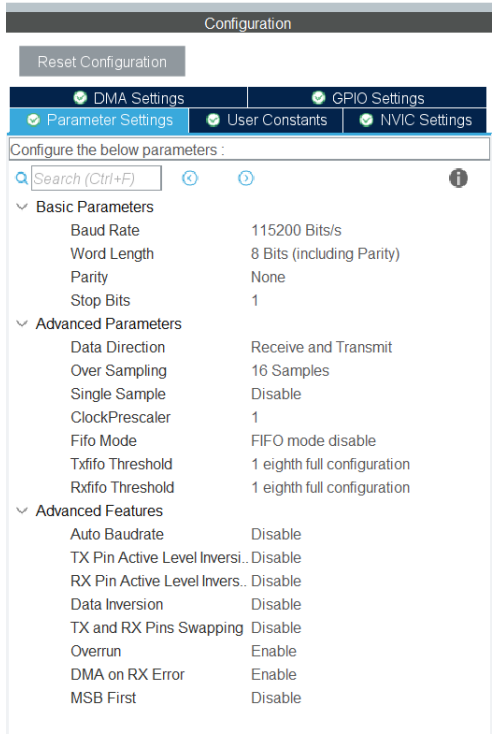
```
  if(HAL_ADCEx_Calibration_Start(&hadc1, ADC_CALIB_OFFSET_LINEARITY, ADC_SINGLE_ENDED) != HAL_OK)
    {
      /* Calibration Error */
      Error_Handler();
    }
```

And finally you can use the code that you wrote in your first post to read the value.

- In Production mode (with linux running):

UART


UART4 Mode and Configuration
Mode
Boot time:
Runtime contexts:
A7 ROM
A7S (TF-A BL2)
A7S (OP-TEE)
A7NS (Linux)
M4 (Cube)
Mode
Asynchronous
Hardware Flow Control (RS232)
Disable
Hardware Flow Control (RS485)


Configuration
Reset Configuration
DMA Settings
GPIO Settings
Parameter Settings
User Constants
NVIC Settings
Configure the below parameters :
Search (Ctrl+F)
Basic Parameters
Baud Rate 115200 Bits/s
Word Length 8 Bits (including Parity)
Parity None
Stop Bits 1
Advanced Parameters
Data Direction Receive and Transmit
Over Sampling 16 Samples
Single Sample Disable
ClockPrescaler 1
Fifo Mode FIFO mode disable
Txfifo Threshold 1 eighth full configuration
Rxfifo Threshold 1 eighth full configuration
Advanced Features
Auto Baudrate Disable
TX Pin Active Level Inversi.. Disable
RX Pin Active Level Invers.. Disable
Data Inversion Disable
TX and RX Pins Swapping Disable
Overrun Enable
DMA on RX Error Enable
MSB First Disable

Problems

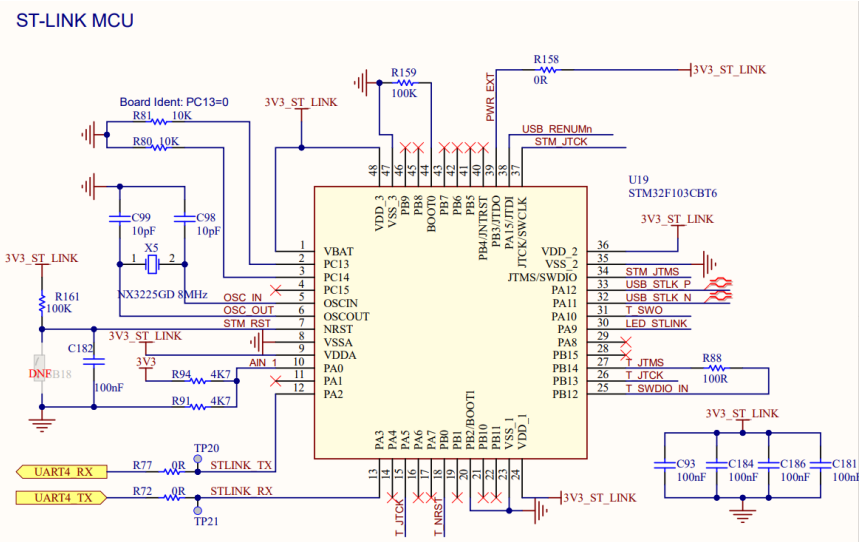
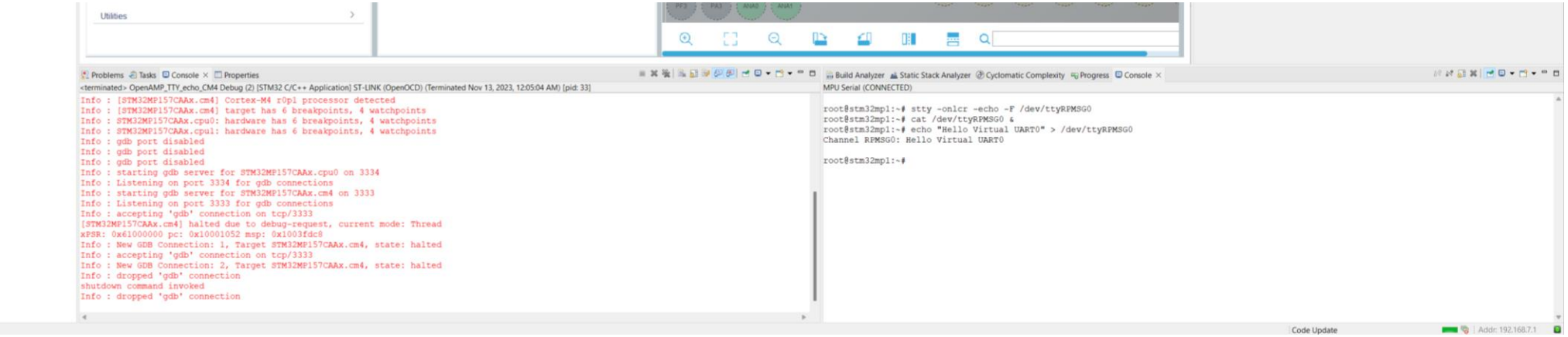


Table 8. Alternate function AF0 to AF7(1)

Port		AF0	AF1	AF2	AF3	AF4	AF5	AF6	AF7
		HDP/SYS/RTC	TIM1/2/16/17/ LPTIM1/SYS/ RTC	SAI1/4/I2C6/ TIM3/4/5/12/ HDP/SYS	SAI4/I2C2/ TIM8/ LPTIM2/3/4/5/ DFSDM1 /SDMMC1	SAI4/ I2C1/2/3/4/5/ USART1/ TIM15/LPTIM2/ DFSDM1/CEC	SPI1/I2S1/ SPI2/I2S2/ SPI3/I2S3/ SPI4/5/6/I2C1/ SDMMC1/3/ CEC	SPI3/I2S3/ SAI1/3/4/ I2C4/UART4/ DFSDM1	SPI2/I2S2/ SPI3/I2S3/ SPI6/ USART1/2/3/6/ UART7/ SDMMC2
Port A	PA0	-	TIM2_CH1/ TIM2_ETR	TIM5_CH1	TIM8_ETR	TIM15_BKIN	-	-	USART2_CTS/ USART2_NSS
	PA1	ETH_CLK	TIM2_CH2	TIM5_CH2	LPTIM3_OUT	TIM15_CH1N	-	-	USART2_RTS/ USART2_DE
	PA2	-	TIM2_CH3	TIM5_CH3	LPTIM4_OUT	TIM15_CH1	-	-	USART2_TX
	PA3	-	TIM2_CH4	TIM5_CH4	LPTIM5_OUT	TIM15_CH2	-	-	USART2_RX
	PA4	HDP0	-	TIM5_ETR	-	SAI4_D2	SPI1_NSS/ I2S1_WS	SPI3_NSS/ I2S3_WS	USART2_CK
	PA5	-	TIM2_CH1/ TIM2_ETR	-	TIM8_CH1N	SAI4_CK1	SPI1_SCK/I2S1 _CK	-	-
	PA6	-	TIM1_BKIN	TIM3_CH1	TIM8_BKIN	SAI4_CK2	SPI1_MISO/ I2S1_SDI	-	-
	PA7	-	TIM1_CH1N	TIM3_CH2	TIM8_CH1N	SAI4_D1	SPI1_MOSI/ I2S1_SDO	-	-
	PA8	MCO1	TIM1_CH1	-	TIM8_BKIN2	I2C3_SCL	SPI3_MOSI/ I2S3_SDO	-	USART1_CK
	PA9	-	TIM1_CH2	-	-	I2C3_SMBA	SPI2_SCK/ I2S2_CK	-	USART1_TX
	PA10	-	TIM1_CH3	-	-	-	SPI3_NSS/ I2S3_WS	-	USART1_RX
	PA11	-	TIM1_CH4	I2C6_SCL	-	I2C5_SCL	SPI2_NSS/ I2S2_WS	UART4_RX	USART1_CTS/ USART1_NSS
	PA12	-	TIM1_ETR	I2C6_SDA	-	I2C5_SDA	-	UART4_TX	USART1_RTS/ USART1_DE

Problems



Olivier GALLIEN
ST Employee

2020-10-27 06:27 AM

Hi @Community member ,

Where did you read that Boot 0 = 0, Boot2 = 1 stand for "Reserved" and Boot0 = 0, Boot2 = 0 for "Engi" ?

Reading here :

https://wiki.st.com/stm32mpu/wiki/STM32MP157x-DKx_-_hardware_description#Boot_related_switches

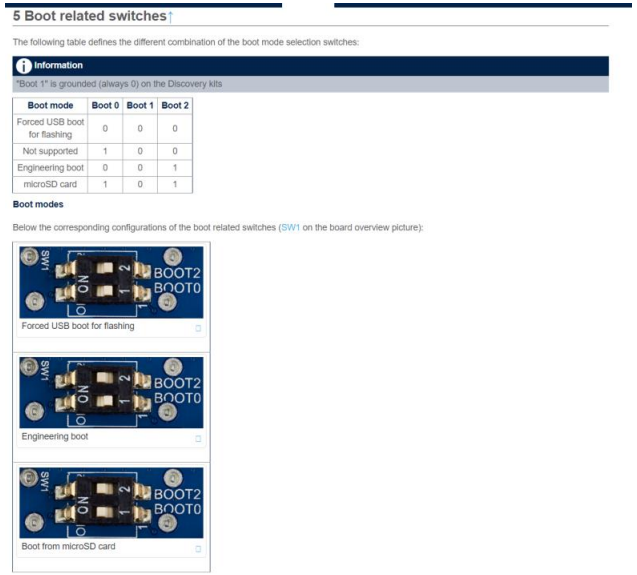
Boot0 = 0, Boot2 = 0 -> Forced USB boot for flashing

Boot0 = 0, Boot2 = 1 -> Engineering boot

Consistent with behavior you noticed.

Olivier

Olivier GALLIEN



Code – Printing ADC values via ST-Link

```
while (1)
{
    HAL_ADC_Start(&hadc2);
    HAL_ADC_PollForConversion(&hadc2, HAL_MAX_DELAY);
    raw = HAL_ADC_GetValue(&hadc2);
    HAL_Delay(10);
    sprintf(msg, "%d\r\n", raw);
    HAL_UART_Transmit(&huart4, msg, strlen(msg), HAL_MAX_DELAY);
    }
```

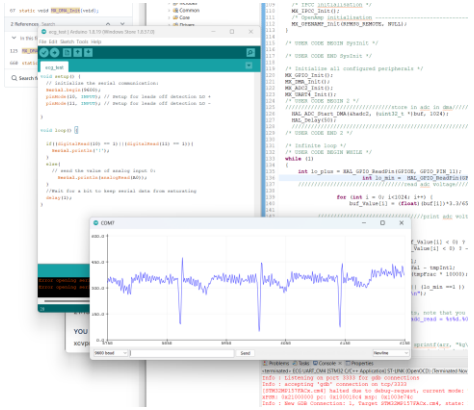
Code – Printing ADC values via ST-Link

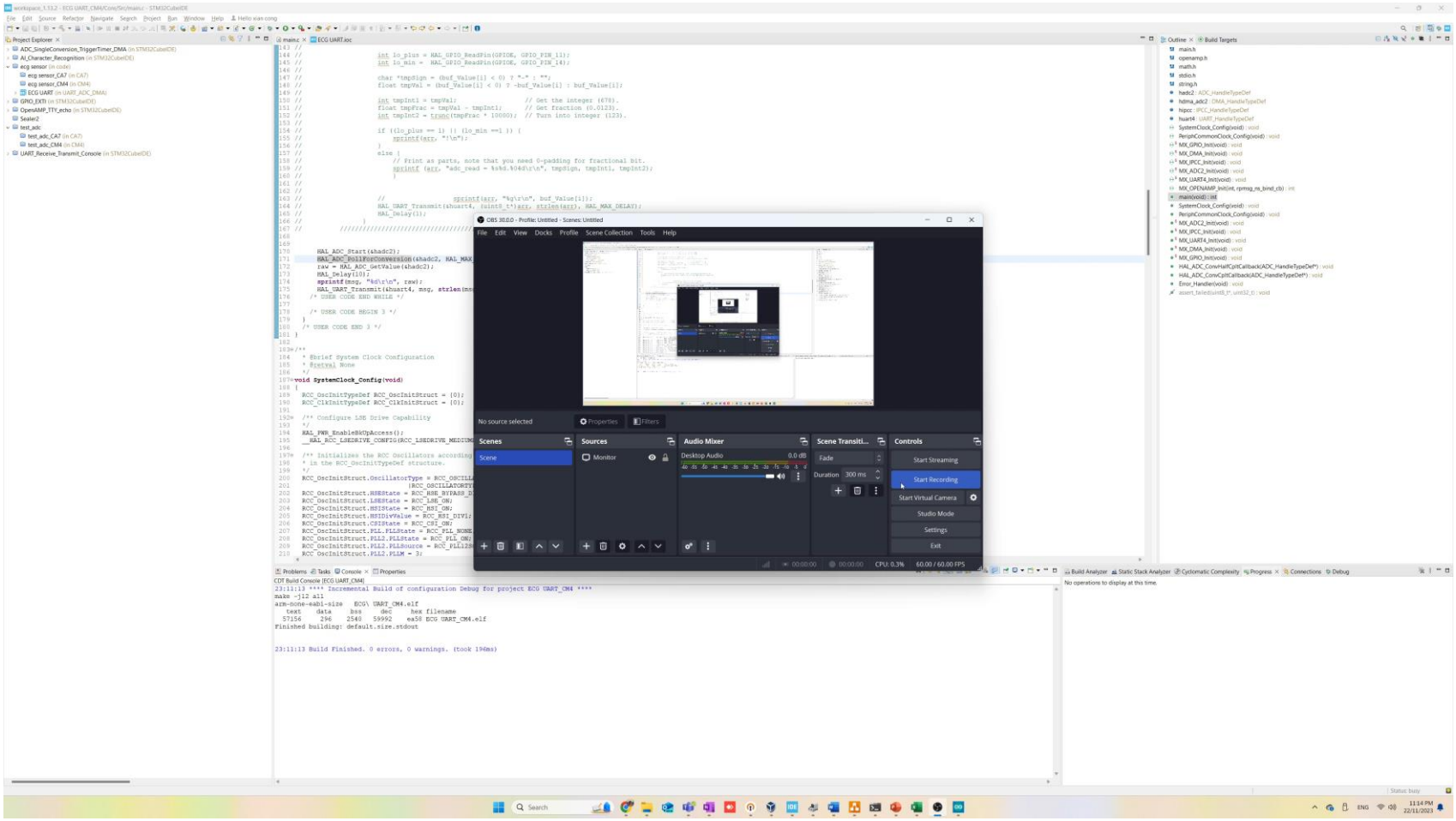
```
__HAL_RCC_VREF_CLK_ENABLE(); // Enable the VREF clock
HAL_SYSCFG_VREFBUF_HighImpedanceConfig(SYSCFG_VREFBUF_HIGH_IMPEDANC
E_DISABLE); // Disable the high impedance mode which is the default
one read page 1694 of refman
HAL_SYSCFG_VREFBUF_VoltageScalingConfig(SYSCFG_VREFBUF_VOLTAGE_SCAL
E1); // To set the volage to 2.5v
HAL_SYSCFG_EnableVREFBUF(); // To enable VREFBUF


if(HAL_ADCEx_Calibration_Start(&hadc2, ADC_CALIB_OFFSET_LINEARITY,
ADC_SINGLE_ENDED) != HAL_OK)
{
/* Calibration Error */
Error_Handler();
}
```

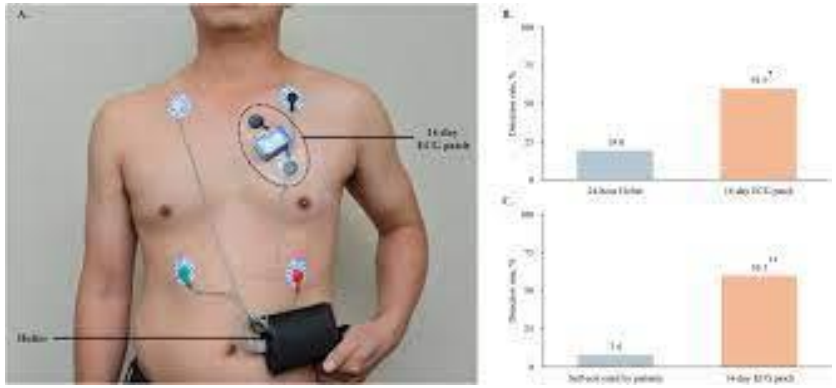
Result

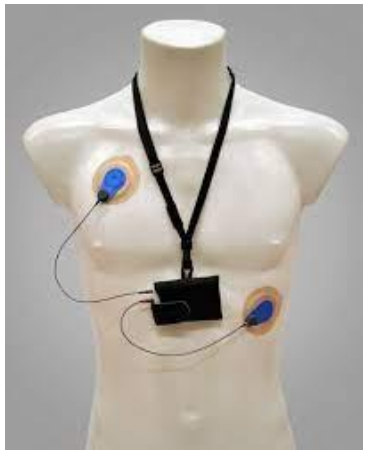
https://youtu.be/zIlDBF6tEII

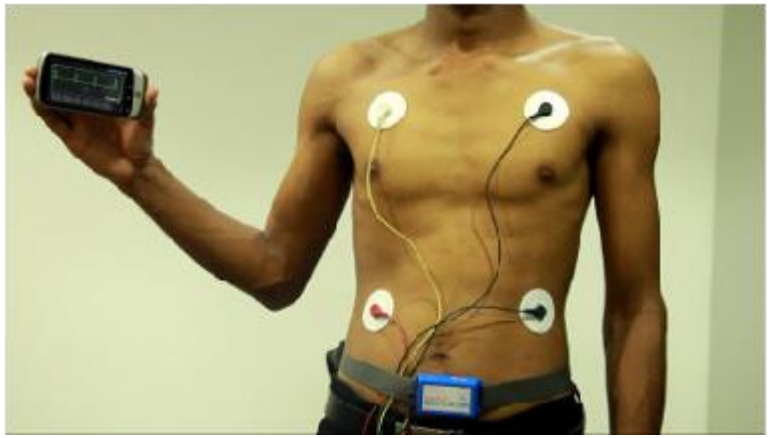




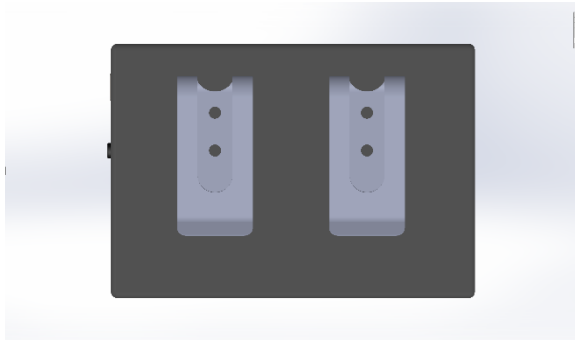
Wearable Design – Reference

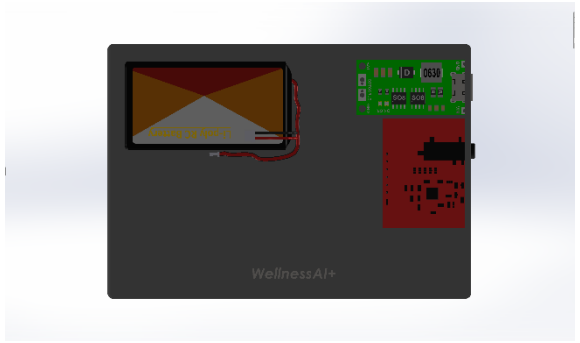


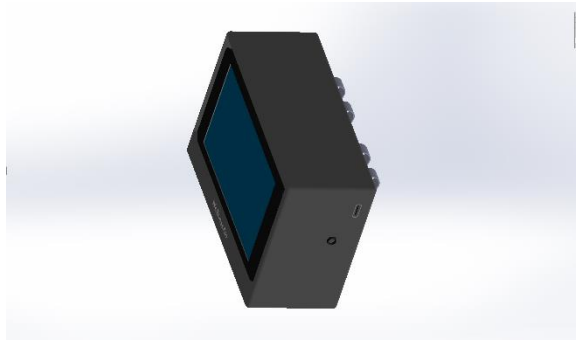


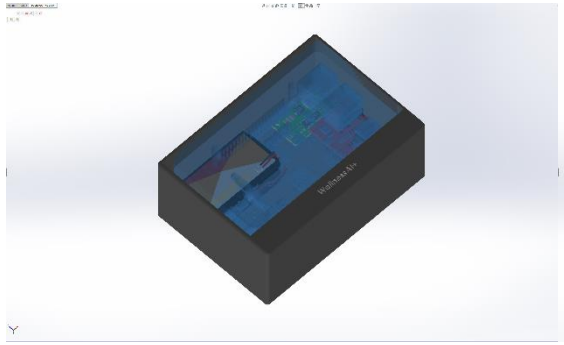


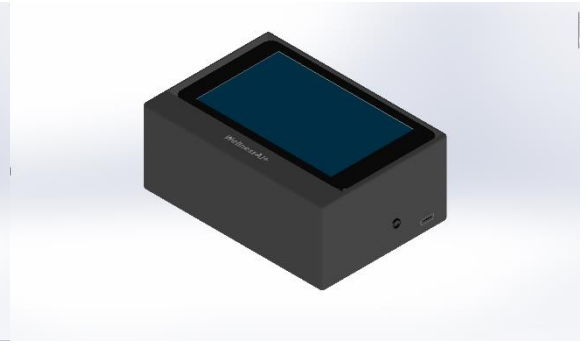
Wearable Design – Clip on Pants

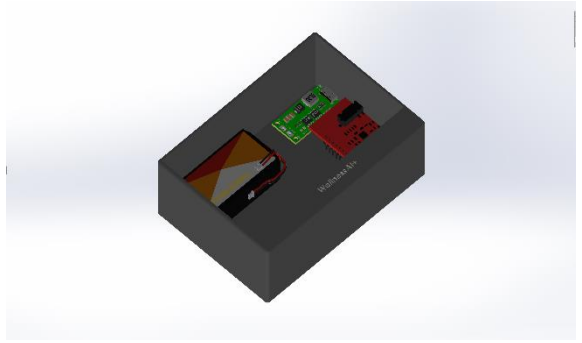










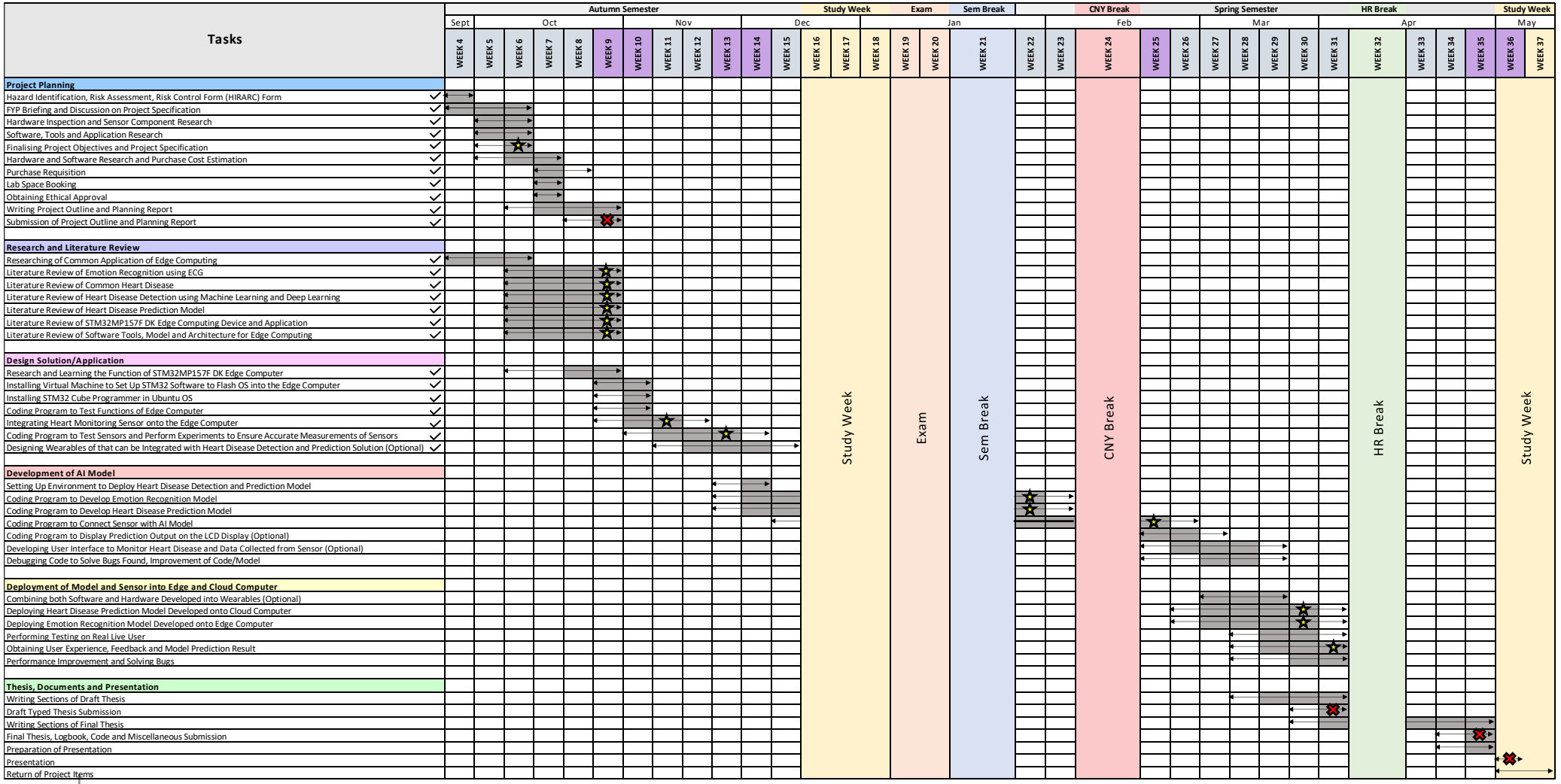


Gantt Chart – What's Next?

Supervisor: Dr Hermawan Nugroho
Moderator: Prof T. Nandha Kumar
Project Code: HN-BEng-23-01

Student Name: Koay Xian Cong
Student ID: 20418760

Legends: Error Bars for Uncertainty: ↔ Moderator Meeting Week:
Submission Dateline: ✖ Project Milestone: ★




University of
Nottingham
UK | CHINA | MALAYSIA