Micro-Controller Experiment

Week5

Teacher: 廖裕評 Yu-Ping Liao

TA: 陳大荃 Da-chuan Chen, 陳恩妮 En-ni Chen

Class Rules

1. No drink besides water.
2. Bring a laptop and breadboard if needed.
3. Ask us TAs to sign and borrow development boards. Do not sign or ask others to sign for you without TAs' permission.
4. Arriving 10 minutes after the bell rings will be regarded as absent.
5. If you damage any borrowed equipment, you have to pay for it.

Homework Rules

1. Includes: A. Class content, B. Class exercise, C. Homework (screenshot or video)
2. Editing software: MS PowerPoint
3. File format: PDF
4. Filename: "date_group_studentID_name.pdf", like "0916_第1組_11028XXX_陳OO.pdf"
5. The homework deadline is 23:59 of the day before the next class. If you are late, then your grade will be deducted.

Contact

If you encounter any problems with this class, please get in touch with us with the following

E-mails:

1. Teacher, Prof. Yu-Ping Liao 廖裕評： lyp@cycu.org.tw
2. TA, Da-chuan Chen 陳大荃： dachuan516@gmail.com
3. TA, En-ni Chen 陳恩妮： anna7125867@gmail.com

Or visit 篤信 Lab353 for further questions.

Outline of the Week

1. I2C introduction
2. I2C Project.
3. Homework 5-1.
4. Homework 5-2.
5. Homework 5-3 Bonus Question.
6. Homework 5-4 Bonus Question.

I2C Introduction

I^2C(Inter-integrated Circuit)

I2C is a synchronous transmission interface for CPU and peripheral chips.

To enable the controller to connect to many low-count peripheral devices with fewer pins.

Only 2 wires are required

SCL(serial clock)&SDA(serial data)

SCL: The line that carries the clock signal.

SDA: The line for the master and slave to send and receive data.

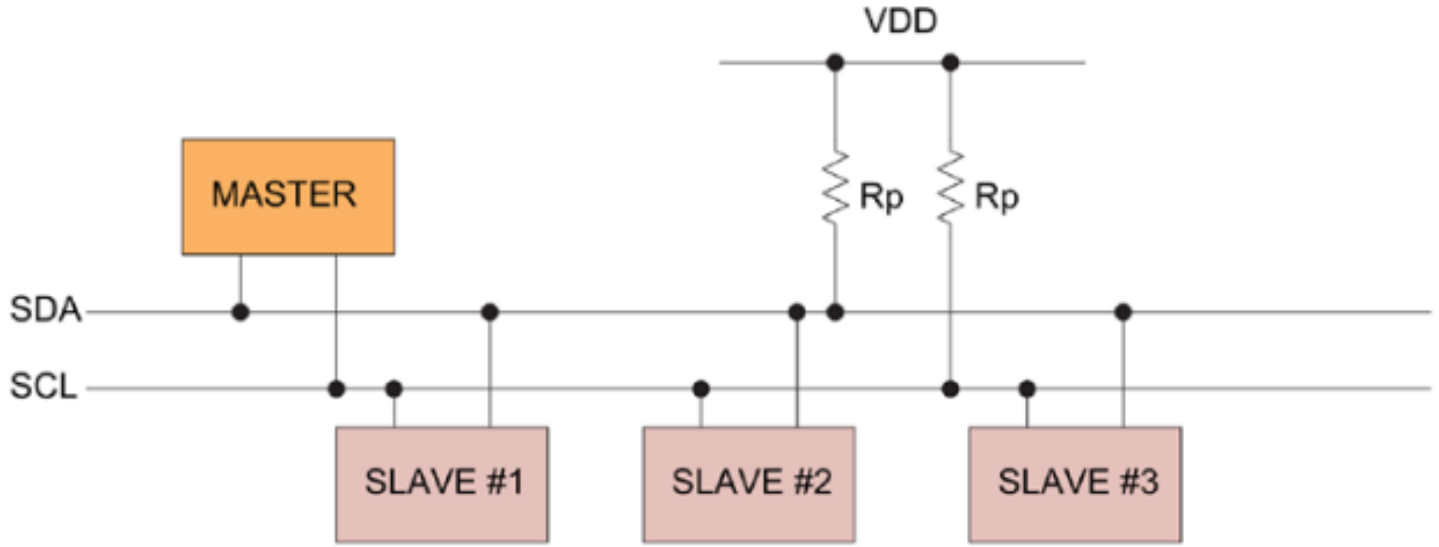


Figure 1: Generalized I^2C Connection Diagram

Data Transfer Rates

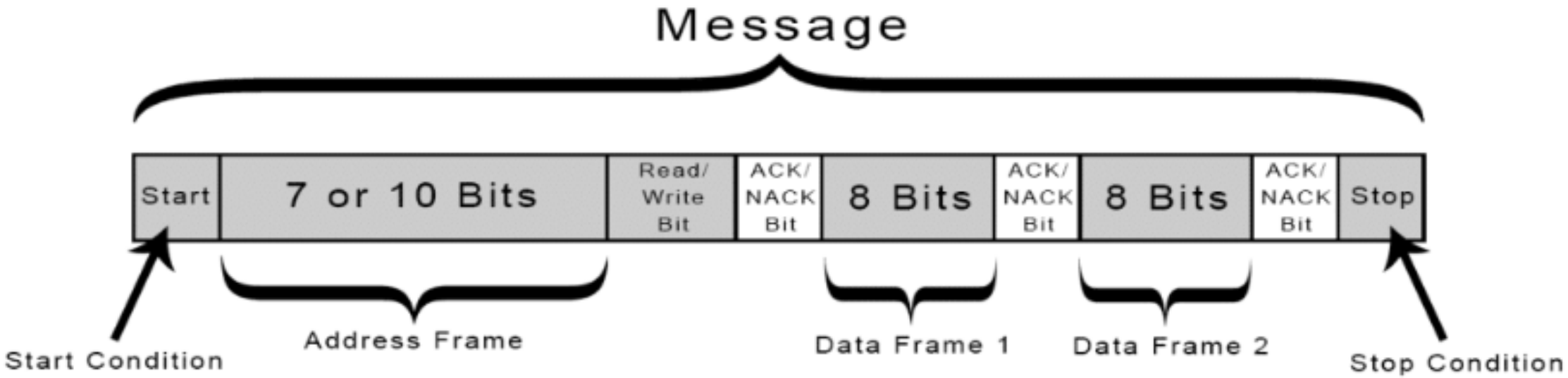
- standard mode (100kMz)
- fast mode (400kMz)
- fast mode plus(1MHz)

➢ Datasheet P41

Note: 1. Guaranteed by design, not tested in production.
2. To achieve 100 kHz standard mode, the peripheral clock frequency must be higher than 2 MHz.
3. To achieve 400 kHz fast mode, the peripheral clock frequency must be higher than 8 MHz.
4. To achieve 1 MHz fast mode plus, the peripheral clock frequency must be higher than 20 MHz.
5. The above characteristic parameters of the I²C bus timing are based on: COMB_FILTER_En = 0 and SEQ_FILTER = 00.
6. The above characteristic parameters of the I²C bus timing are based on: COMB_FILTER_En = 1 and SEQ_FILTER = 00.

How I^2C work?

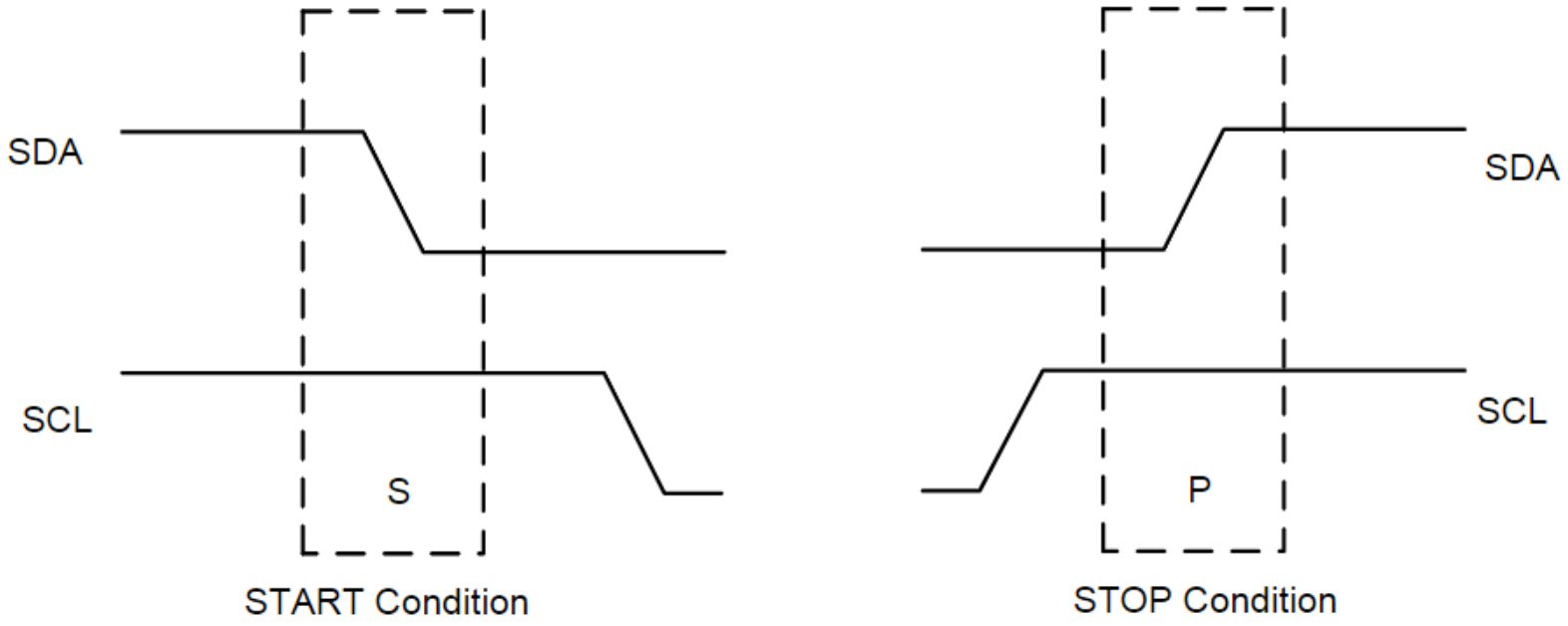
Start=>Address=>Read/Write=>Ack=>Data=>Stop



https://www.circuitbasics.com/basics-of-the-i2c-communication-protocol/

Start & Stop Condition

- User Manual P448



Data Validity

- User Manual P448

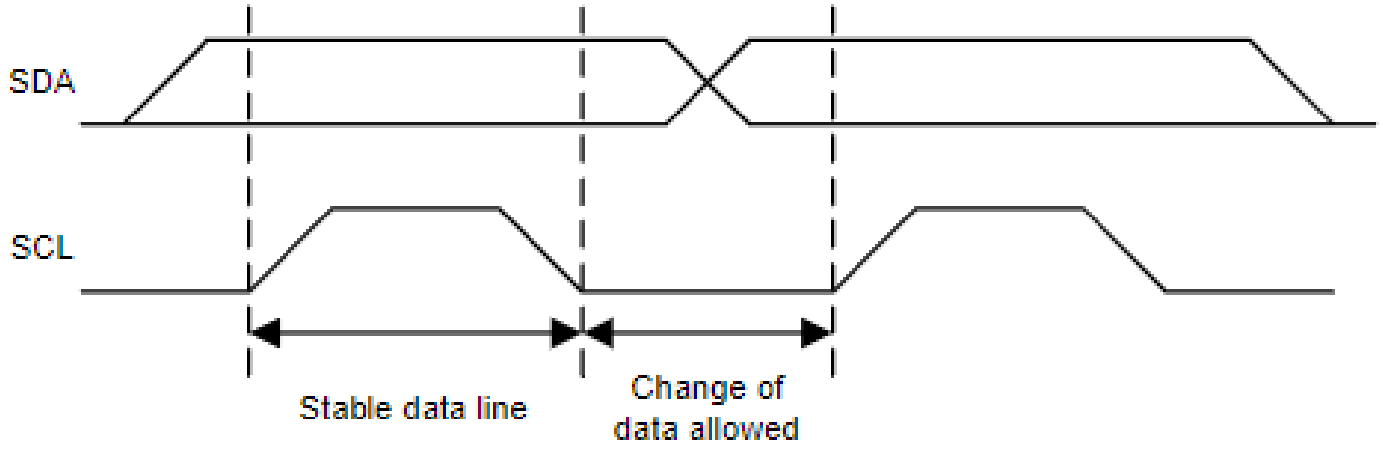


Figure 138. Data Validity

7-bit Addressing Mode

- User Manual P449

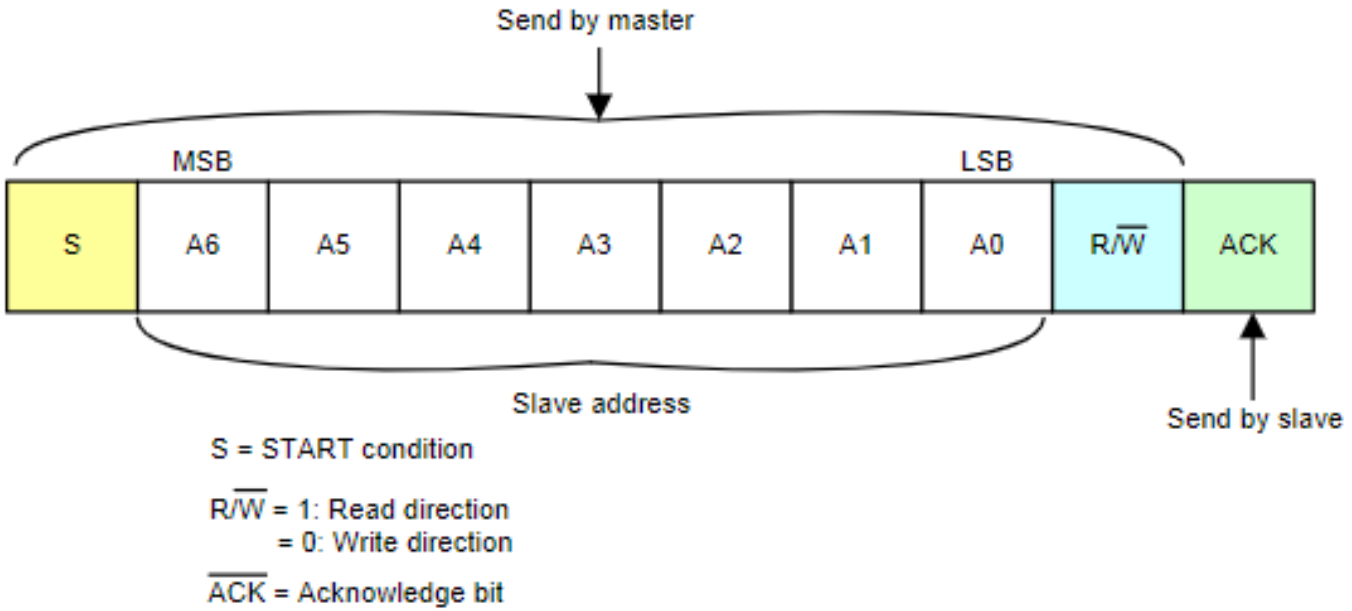
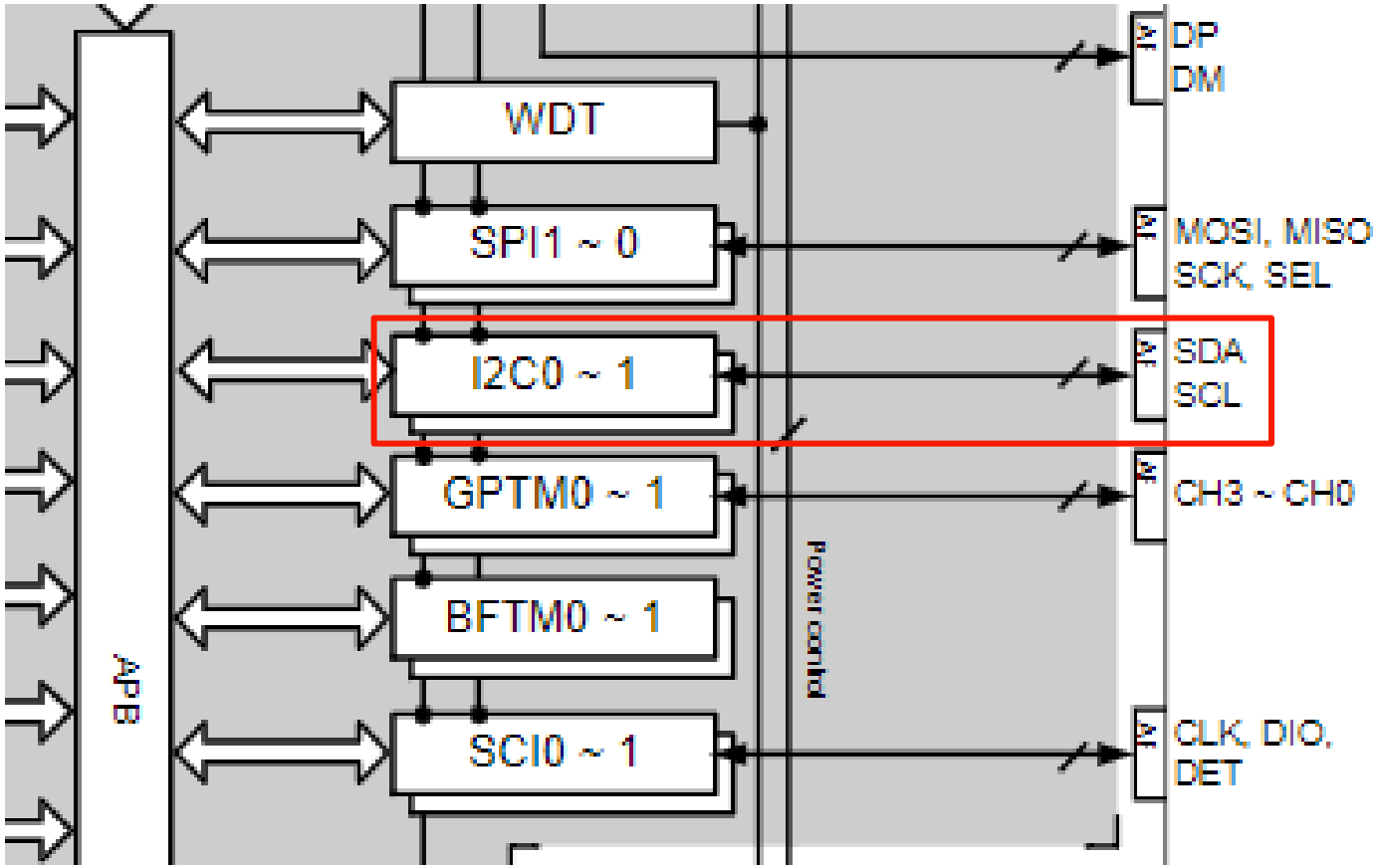


Figure 139. 7-bit Addressing Mode

7-bit Addressing Mode

- Datasheet P19

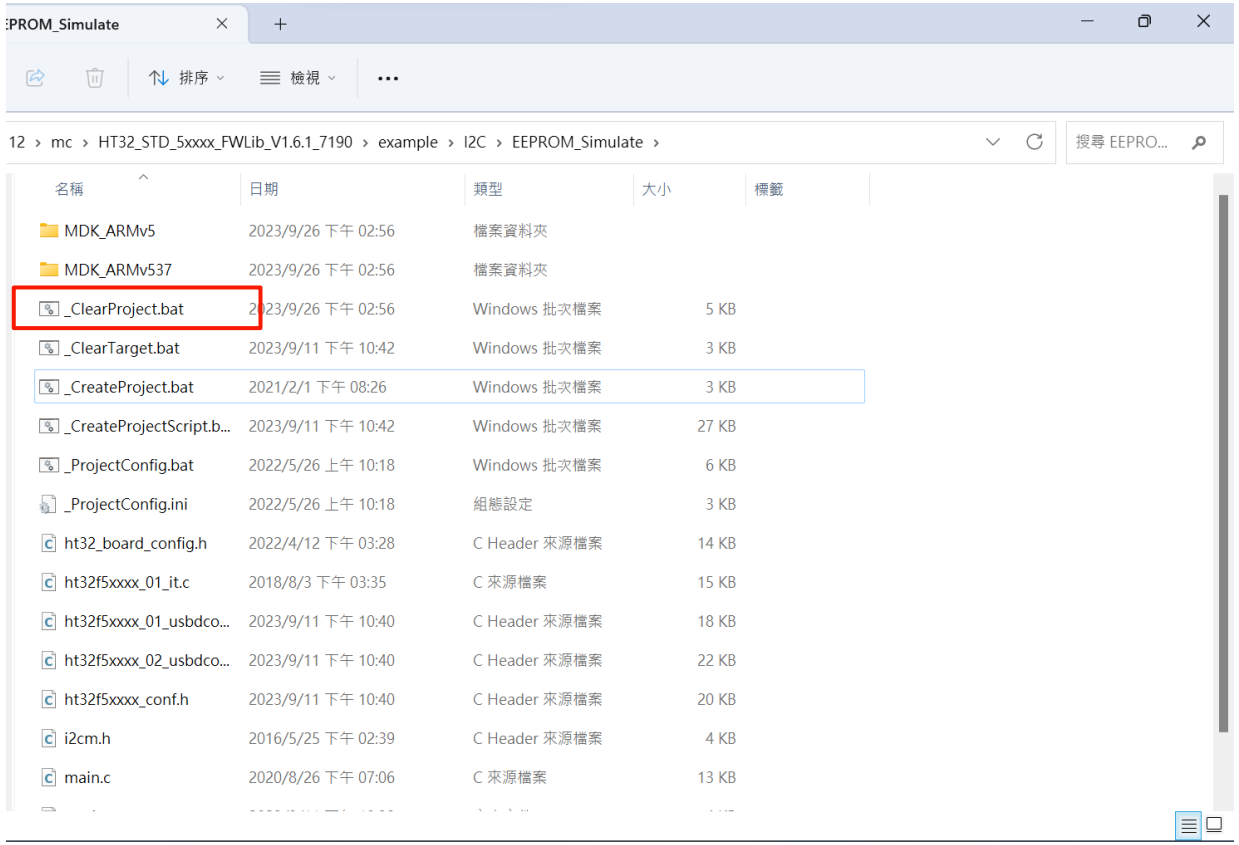


I2C Project

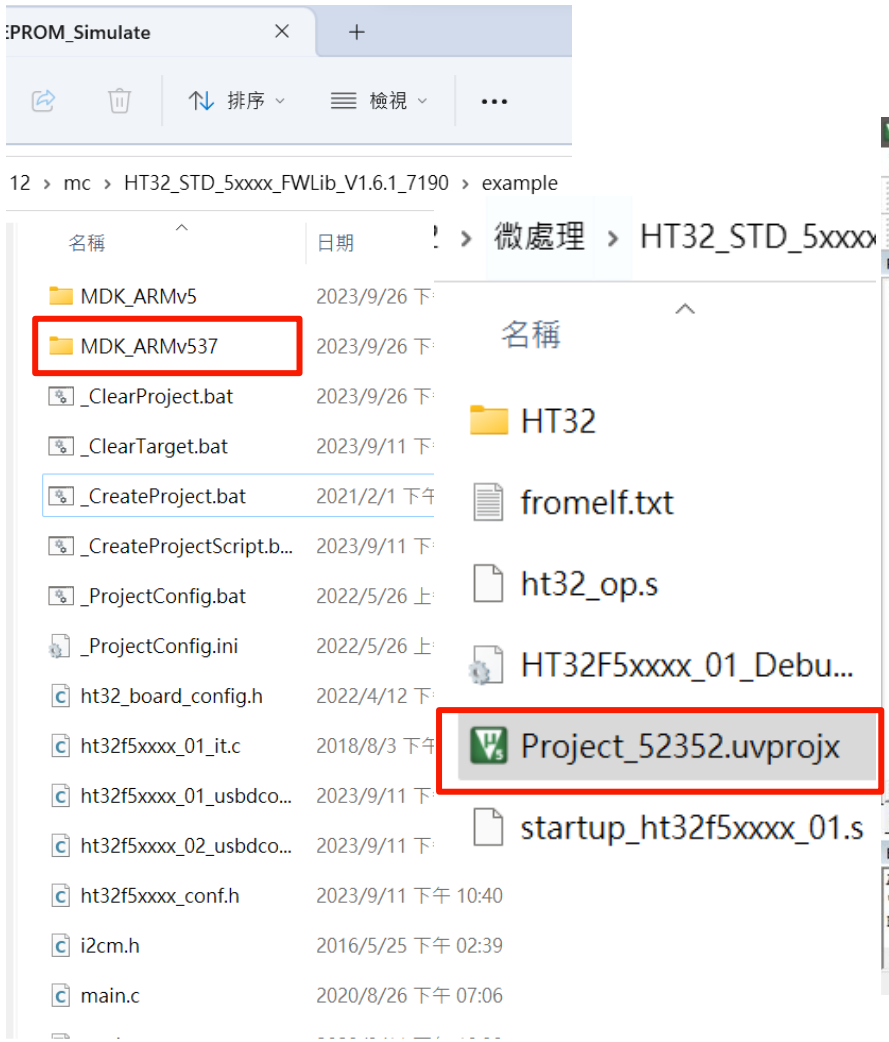
1. Execute “_CreatProject”

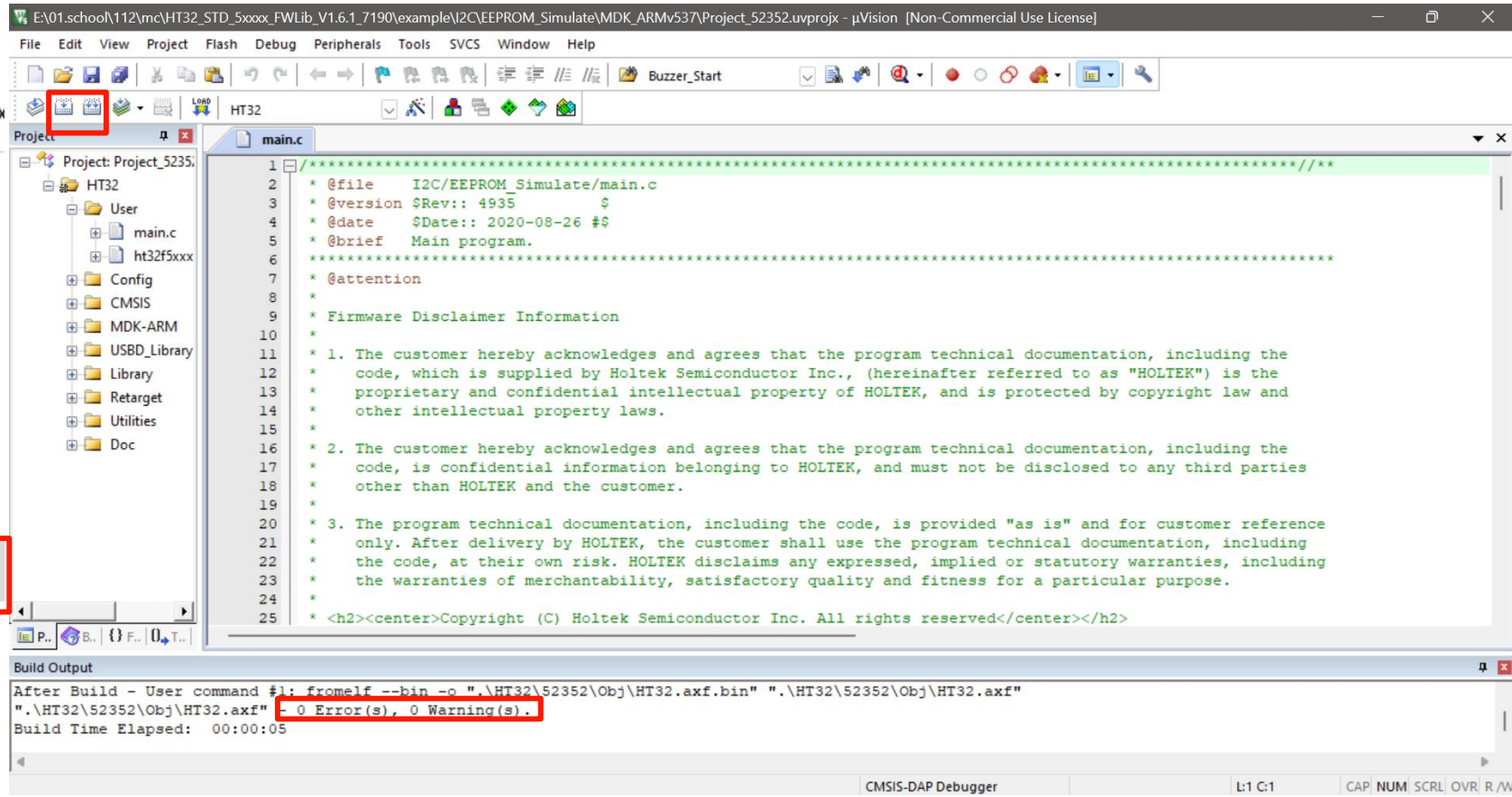
1. Go to “~/HT32_STD_5xxxx_FWLib_V1.5.1_7084/example/I2C/EEPROM_Simulate”.

2. Double click “_CreateProject.bat”.



2. Launch project





main

```
66  int main(void)
67  {
68    /* Initialize LED1 & LED2 on HT32 board
69    HT32F_DVB_LEDInit(HT_LED1);
70    HT32F_DVB_LEDInit(HT_LED2);
71
72    /*Init Key 1
73    HT32F_DVB_PBInit(BUTTON_KEY1, BUTTON_MODE_GPIO);
74
75  #ifdef LOOP_BACK
76    /* Configure I2C Slave as an EEPROM simulation
77    I2CS_EEPROMsim_Init();
78
79    /* Configure I2C Master
80    I2CM_Init();
81
82    EEPROM_WriteReadTest();
```

LED Initialization Setting

Slave EEPROM Setting

Master Setting

I2C test

```
 98  #endif
 99
100    HT32F_DVB_LEDOn(HT_LED1);
101
102    /* Infinite loop
103    while (1);
104  }
105
```

Turn on LED1

I2CS_EEPROM_Init

```
110  void I2CS_EEPROMsim_Init(void)
111  {
112    { /* Enable peripheral clock
113      CKCU_PeripClockConfig_TypeDef CKCUClock = {{0}};
114      CKCUClock.Bit.AFIO = 1;
115      HTCFG_I2C_EEPROM_CLK(CKCUClock)    = 1;
116      CKCU_PeripClockConfig(CKCUClock, ENABLE);
117    }
118
119    /* Configure SDA and SCL to I2C1 mode
120    AFIO_GPxConfig(HTCFG_I2C_EEPROM_SCL_GPIO_ID, HTCFG_I2C_EEPROM_SCL_AFIO_PIN, AFIO_FUN_I2C);
121    AFIO_GPxConfig(HTCFG_I2C_EEPROM_SDA_GPIO_ID, HTCFG_I2C_EEPROM_SDA_AFIO_PIN, AFIO_FUN_I2C);
```

CKCU Setting

I2C Slave's Pin Setting

- Datasheet P27

Table 3. Pin Assignment for 33-pin QFN, 48/64-pin LQFP Packages

Package			Alternate Function Mapping														
			AF0	AF1	AF2	AF3	AF4	AF5	AF6	AF7	AF8	AF9	AF10	AF11	AF12	AF13	AF14	AF15
64 LQFP	48 LQFP	33 QFN	System Default	GPIO	ADC	CMP	MCTM /GPTM	SPI	USART /UART	I²C	SCI	EBI	I2S	N/A	N/A	SCTM	N/A	System Other
1	1	1	PA0		ADC_ IN0		GT1_ CH0	SPI1_ SCK	USR0_ RTS	I2C1_ SCL	SCI0_ CLK		I2S_WS					
2	2	2	PA1		ADC_ IN1		GT1_ CH1	SPI1_ MOSI	USR0_ CTS	I2C1_ SDA	SCI0_ DIO		I2S_ BCLK					
3	3	3	PA2		ADC_ IN2		GT1_ CH2	SPI1_ MISO	USR0_ TX				I2S_ SDO					
4	4	4	PA3		ADC_ IN3		GT1_ CH3	SPI1_ SEL	USR0_ RX				I2S_SDI					
5	5	5	PA4		ADC_ IN4		GT0_ CH0	SPI0_ SCK	USR1_ TX	I2C0_ SCL	SCI1_ CLK							
6	6	6	PA5		ADC_ IN5		GT0_ CH1	SPI0_ MOSI	USR1_ RX	I2C0_ SDA	SCI1_ DIO							

I2CS_EEPROM_Init

```
124  { /* EEPROM configuration                                                    */
125
126    /* !!! NOTICE !!!
127       Notice that the local variable (structure) did not have an initial value.
128       Please confirm that there are no missing members in the parameter settings below in this function.
129    */
130    I2C_InitTypeDef I2C_InitStructure;
131
132    I2C_InitStructure.I2C_GeneralCall = DISABLE;
133    I2C_InitStructure.I2C_AddressingMode = I2C_ADDRESSING_7BIT;
134    I2C_InitStructure.I2C_Acknowledge = ENABLE;
135    I2C_InitStructure.I2C_OwnAddress = EEPROM_ADDRESS;
136    I2C_InitStructure.I2C_Speed = 400000;
137    I2C_InitStructure.I2C_SpeedOffset = 0;
138    I2C_Init(HTCFG_I2C_EEPROM_PORT, &I2C_InitStructure);
139  }
140
141  /* Enable EEPROM interrupts                                                 */
142  I2C_IntConfig(HTCFG_I2C_EEPROM_PORT, I2C_INT_ADRS | I2C_INT_RXDNE | I2C_INT_TXDE | I2C_INT_RXNACK | I2C_INT_STO, ENABLE);
143
144  /*Enable NVIC EEPROM interrupt                                             */
145  NVIC_EnableIRQ(HTCFG_I2C_EEPROM_IRQn);
146
147  /* Enable EEPROM                                                            */
148  I2C_Cmd(HTCFG_I2C_EEPROM_PORT, ENABLE);
149 }
```

7bits Register

Slave Address

Transfer Rates

I2C Interrupts Setting

I2CM_Init

```
170     /* !!! NOTICE !!!
171        Notice that the local variable (structure) did not have an initial value.
172        Please confirm that there are no missing members in the parameter settings below in this function.
173     */
174     I2C_InitTypeDef  I2C_InitStructure;
175
176     I2C_InitStructure.I2C_GeneralCall      = DISABLE;
177     I2C_InitStructure.I2C_AddressingMode   = I2C_ADDRESSING_7BIT;
178     I2C_InitStructure.I2C_Acknowledge      = DISABLE;
179     I2C_InitStructure.I2C_OwnAddress       = 0x7F;
180     I2C_InitStructure.I2C_Speed            = 400000;
181     I2C_InitStructure.I2C_SpeedOffset      = 0;
182     I2C_Init(HTCFG_I2C_MASTER_PORT, &I2C_InitStructure);
183   }
184
185   /* Enable I2C interrupts                                                                                    */
186   I2C_IntConfig(HTCFG_I2C_MASTER_PORT, I2C_INT_STA | I2C_INT_ADRS | I2C_INT_RXDNE | I2C_INT_TXDE
187                 | I2C_INT_ARBLOS | I2C_INT_RXNACK | I2C_INT_BUSERR | I2C_INT_TOUT , ENABLE);
```

7bits Register

Master Address

Transfer Rates

EEPROM_WriteReadTest

```
252  void EEPROM_WriteReadTest(void)
253  {
254    if (I2CM_BufferWrite(EEPROM_ADDRESS, 0x00, (u8*)TestData, 16) != I2CM_OK)
255    {
256      /* Write fail
257      HT32F_DVB_LEDOn(HT_LED2);
258      while (1);
259    }
260    if (I2CM_BufferRead(EEPROM_ADDRESS, 0x00, ReadBuffer, 16) != I2CM_OK)
261    {
262      /* Read fail
263      HT32F_DVB_LEDOn(HT_LED2);
264      while (1);
265    }
266  }
```

If I2C write fail, LED2 turn on.

If I2C read fail, LED2 turn on.

I2CM_BufferRead

```
200  u32 I2CM_BufferRead(u16 dev, u8 word_addr, u8* data, u16 count)
201  {
202    I2CM_Transfer.RegAddr = word_addr;
203    I2CM_Transfer.Buffer      = data;
204    I2CM_Transfer.Length      = count;
205    I2CM_Transfer.Counter     = 0;
206    I2CM_Transfer.Direction   = I2CM_DIRECTION_IN;
207    I2CM_Transfer.DevAddr     = dev;
208    I2CM_Transfer.Locked      = TRUE;
209    I2CM_Transfer.RetryDownCounter = I2CM_BUS_MAX_RETRY;
210
211    /* Send I2C START
212    I2C_TargetAddressConfig(HTCFG_I2C_MASTER_PORT, I2CM_Transfer.DevAddr, I2C_MASTER_WRITE);
213
214    while (I2CM_Transfer.Locked);
215    while (I2C_GetFlagStatus(HTCFG_I2C_MASTER_PORT, I2C_FLAG_BUSBUSY));
216
217    I2CM_Transfer.Buffer = NULL;
218
219    return (u32)I2CM_Transfer.Status;
220  }
```

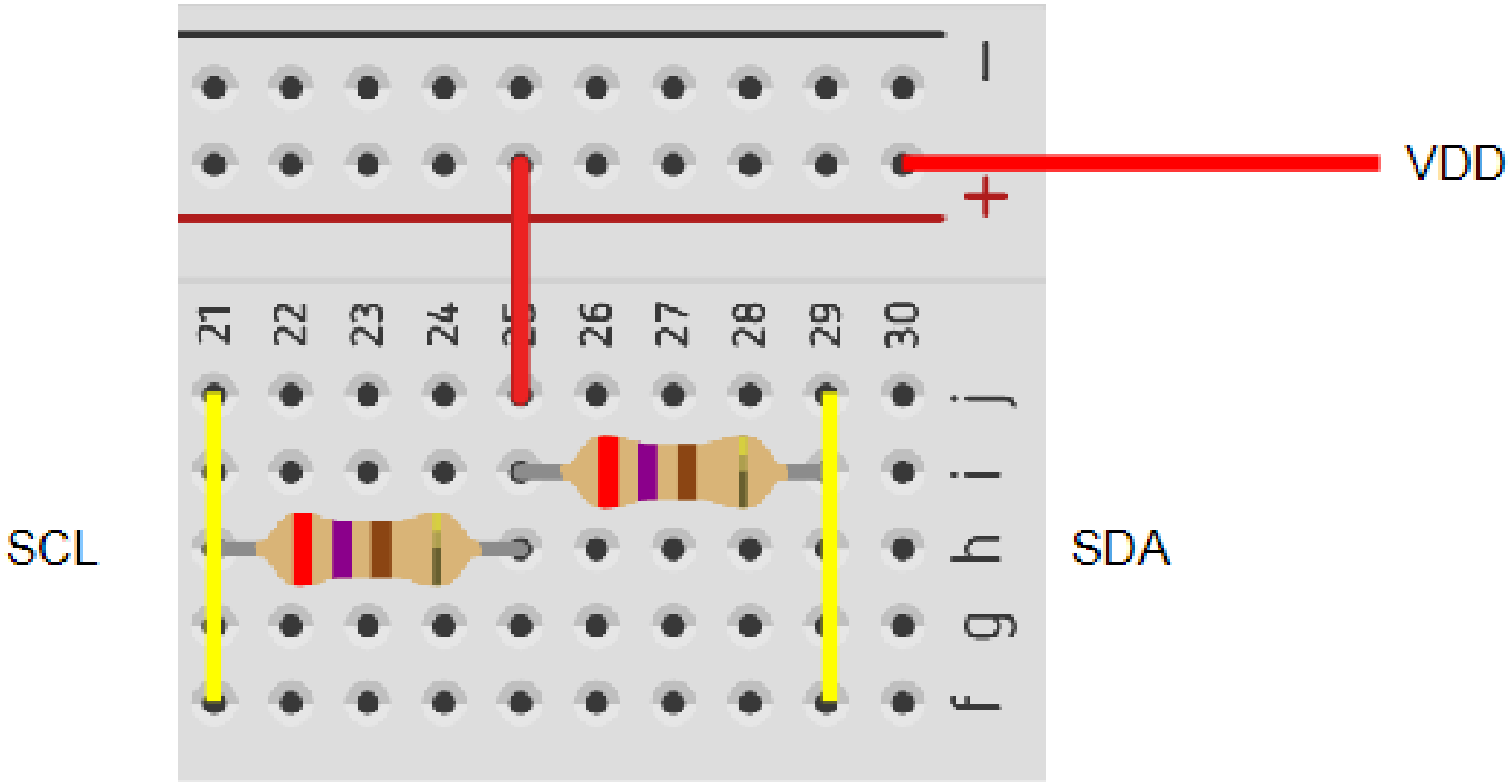
Setting

I2CM_BufferWrite

```
226  u32 I2CM_BufferWrite(u16 dev, u8 word_addr, u8* data, u16 count)
227  {
228    I2CM_Transfer.RegAddr      = word_addr;
229    I2CM_Transfer.Buffer       = data;
230    I2CM_Transfer.Length       = count;
231    I2CM_Transfer.Counter      = 0;
232    I2CM_Transfer.Direction    = I2CM_DIRECTION_OUT;
233    I2CM_Transfer.DevAddr      = dev;
234    I2CM_Transfer.Locked       = TRUE;
235    I2CM_Transfer.RetryDownCounter = I2CM_BUS_MAX_RETRY;
236
237    /* Send I2C START
238    I2C_TargetAddressConfig(HTCFG_I2C_MASTER_PORT, I2CM_Transfer.DevAddr, I2C_MASTER_WRITE);
239
240    while (I2CM_Transfer.Locked);
241    while (I2C_GetFlagStatus(HTCFG_I2C_MASTER_PORT, I2C_FLAG_BUSBUSY));
242
243    I2CM_Transfer.Buffer = NULL;
244
245    return (u32)I2CM_Transfer.Status;
246  }
```

Setting

Wiring



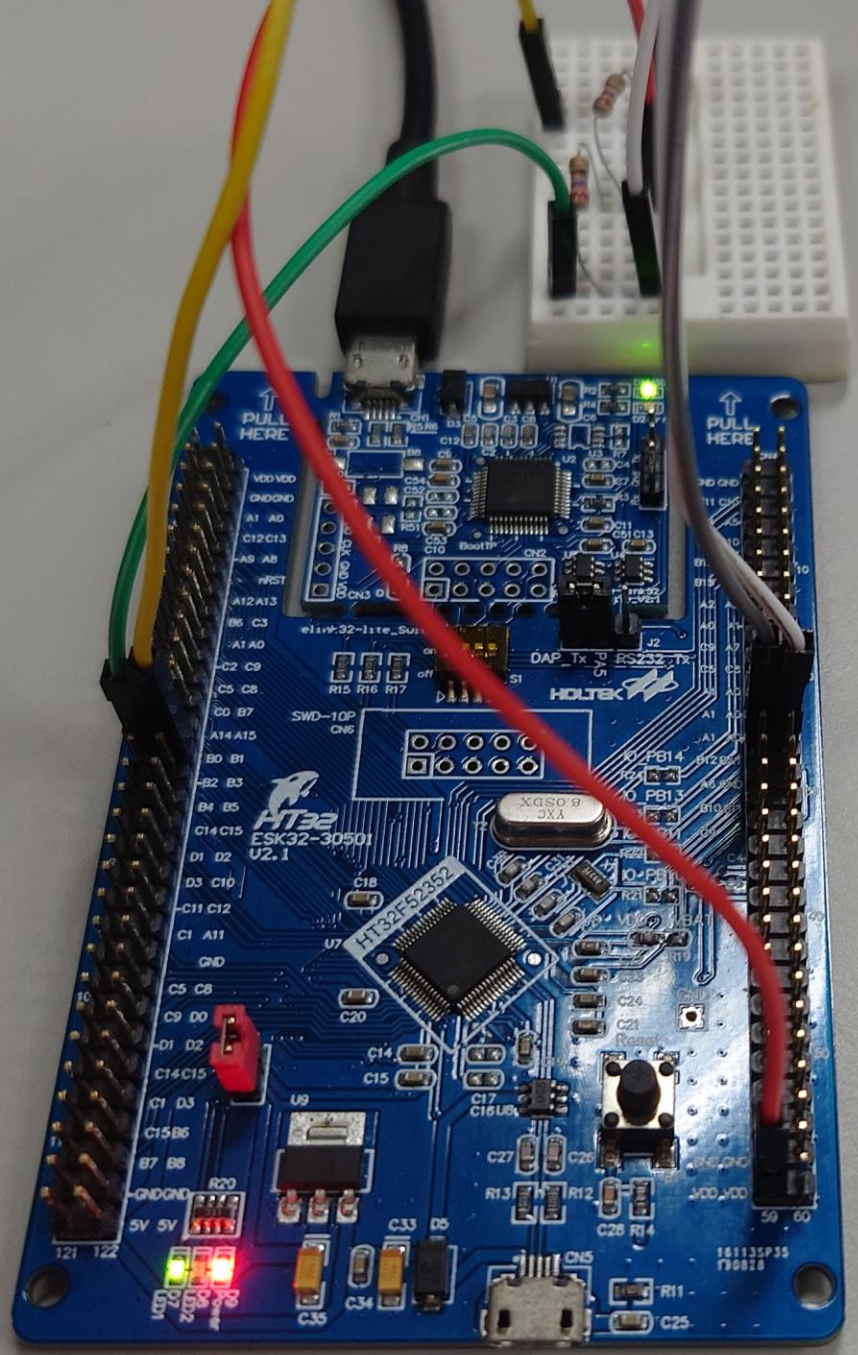
Homework W5-1.

https://github.com/CYCU-AIoT-System-Lab/Microcontroller-Experiment/blob/main/w5/I2C-EEPROM_Simulate-Experiment_Steps.md

Execute the example

- Objective: Execute example project.
- Hint:

1. Use key “F12” to locate specified pins and connect them.
2. Compile, download, execute example project.



https://youtu.be/DjRtjLKtLHk

Homework W5-2

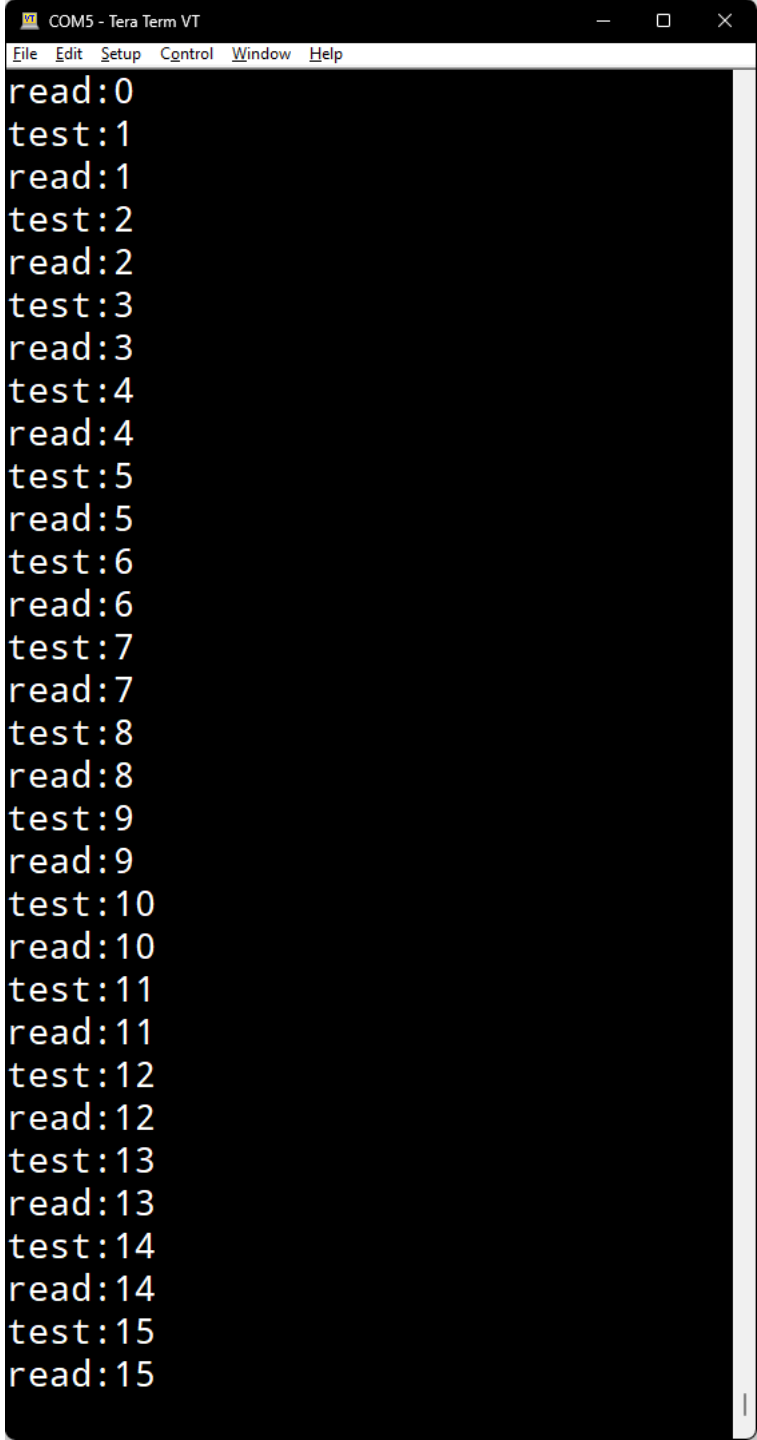
Show the result on TeraTerm

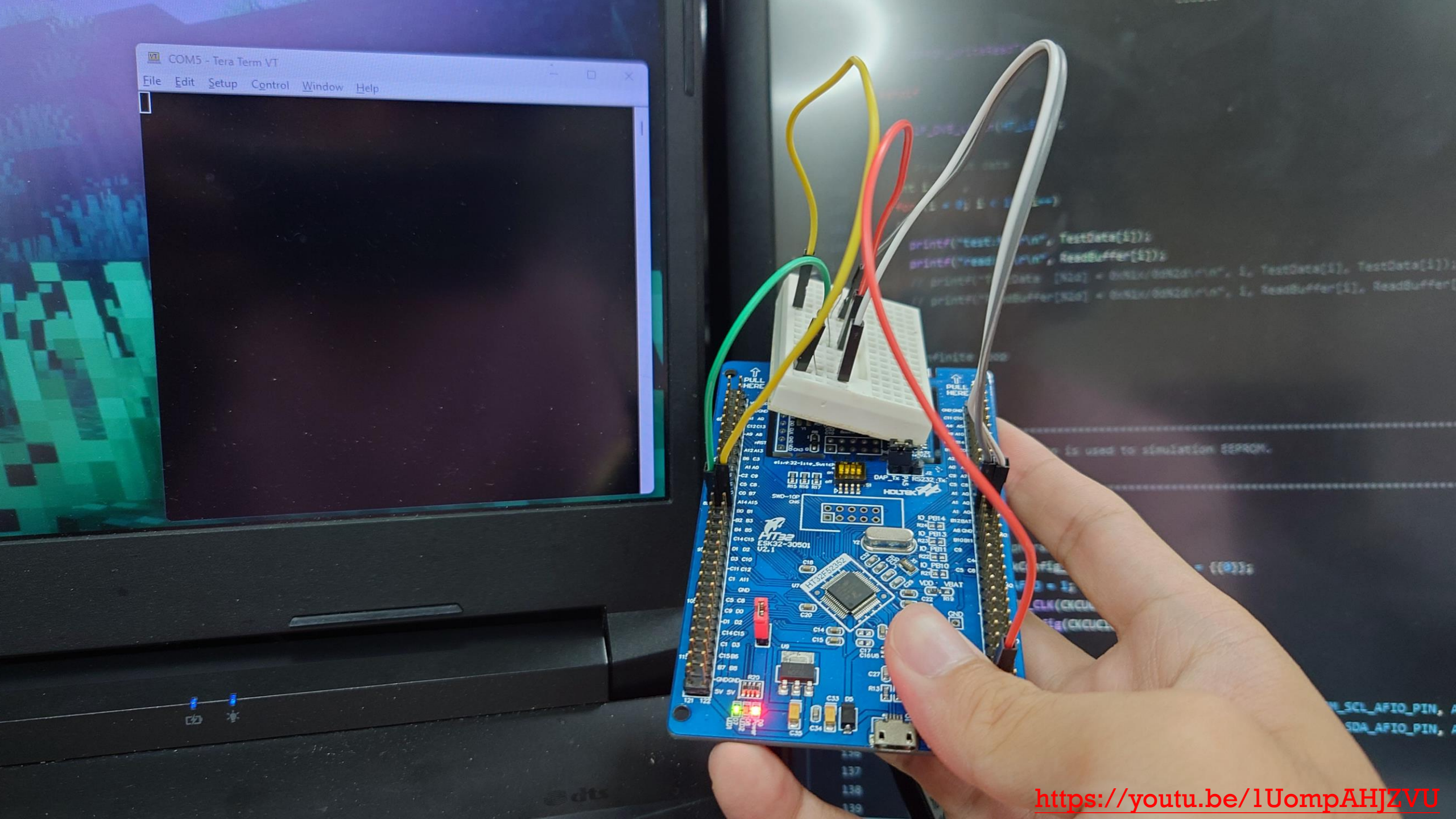
- Objective: Display I2C transfer result with Tera Term.
- Hint:

1. Add following two functions at appropriate location.

1
```
for (i=0;i<16;i++){
    printf("test : %d\r\n", TestData[i]);
    printf("read : %d\r\n", ReadBuffer[i]);
  }
```

2
```
RETARGET_Configuration();
```



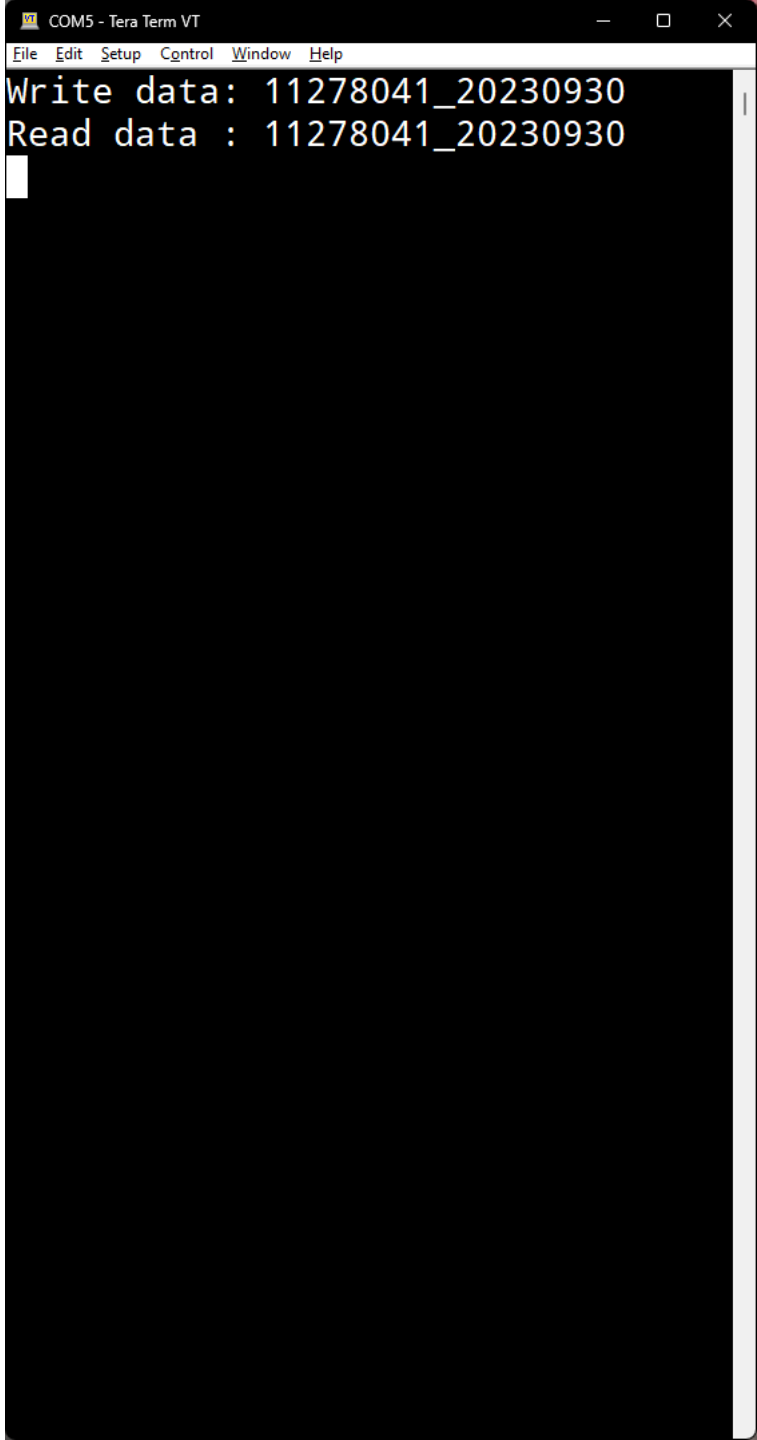

COM5 - Tera Term VT
File Edit Setup Control Window Help
HT32F52352
ESK32-30501
V2.1
https://youtu.be/1UompAHJZVU

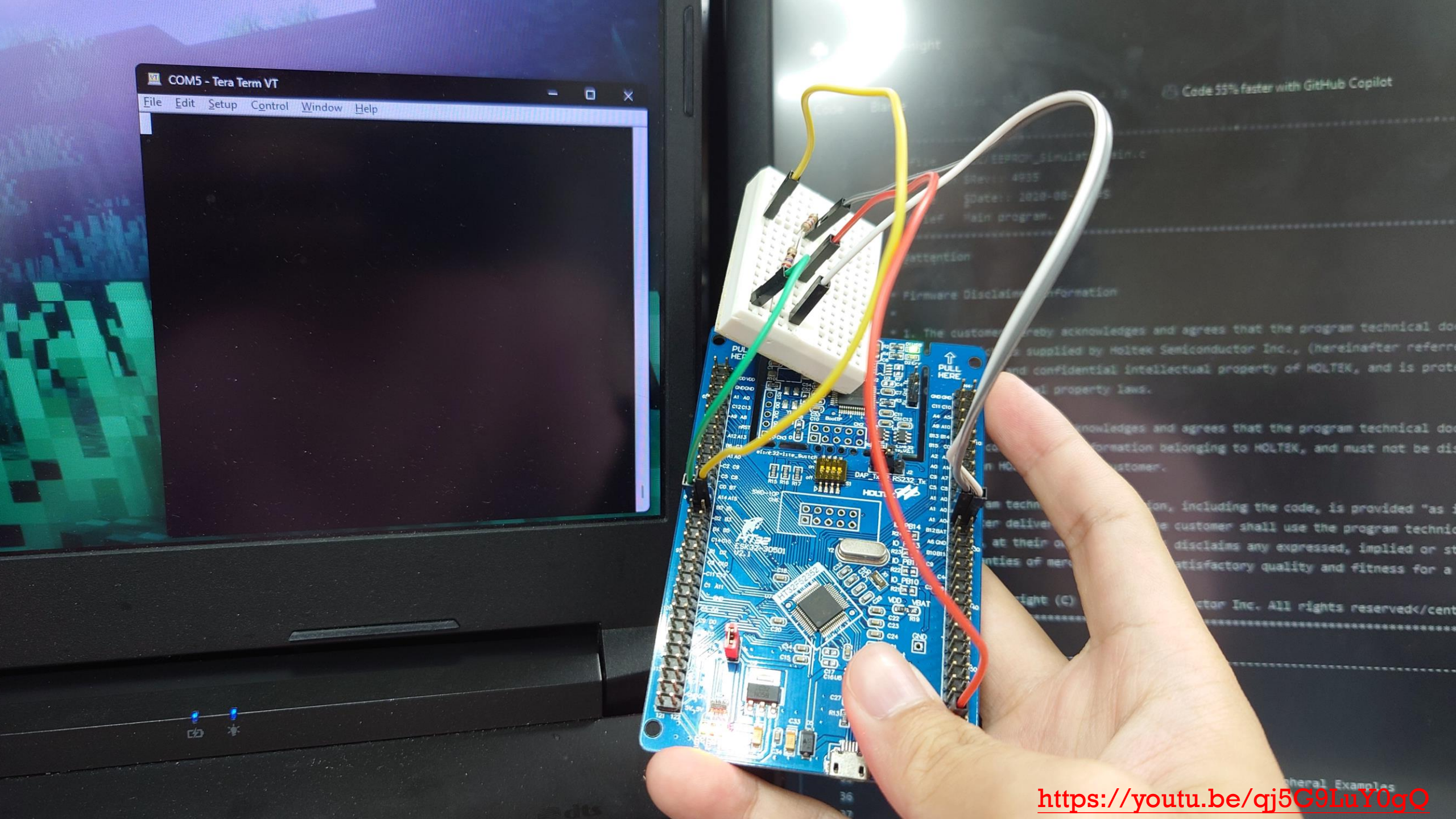
Homework W5-3 Bonus Question

Transfer and show your student number

- Objective: Display "<student_number>_<date>" and explain how your code work.
- Hint:

1. Modify variable "TestData". Only numbers should be included.
2. Print out the result by using combination of array elements.




COM5 - Tera Term VT
File Edit Setup Control Window Help
Code 55% faster with GitHub Copilot
https://youtu.be/qj5G9LuY0gQ

Homework W5-4 Bonus Question

Explain code

- Objective: Explain what the following two lines of code means.

```
/* Enable EEPROM interrupts                                                                    */
I2C_IntConfig(HTCFG_I2C_EEPROM_PORT, I2C_INT_ADRS | I2C_INT_RXDNE | I2C_INT_TXDE | I2C_INT_RXNACK | I2C_INT_STO, ENABLE);
```

```
/* Enable I2C interrupts                                                                       */
I2C_IntConfig(HTCFG_I2C_MASTER_PORT, I2C_INT_STA | I2C_INT_ADRS | I2C_INT_RXDNE | I2C_INT_TXDE
                  | I2C_INT_ARBLOS | I2C_INT_RXNACK | I2C_INT_BUSERR | I2C_INT_TOUT , ENABLE);
```

Class Dismissed