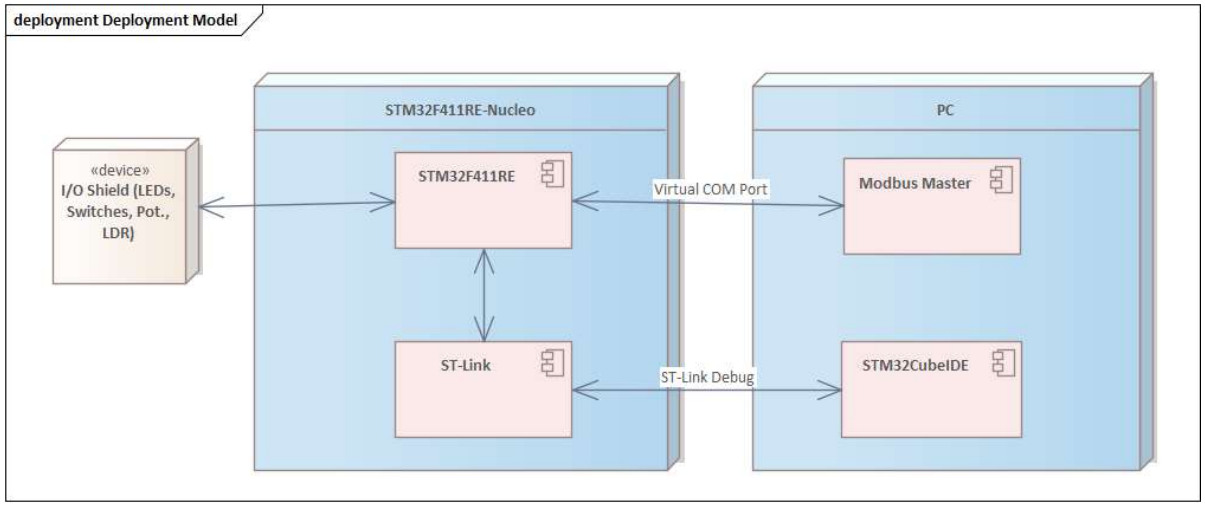
Embedded Systems: Learning by Doing

Our Platform in this Workshop

- STM32F411RE Nucleo-32
- PC
- I/O Shield

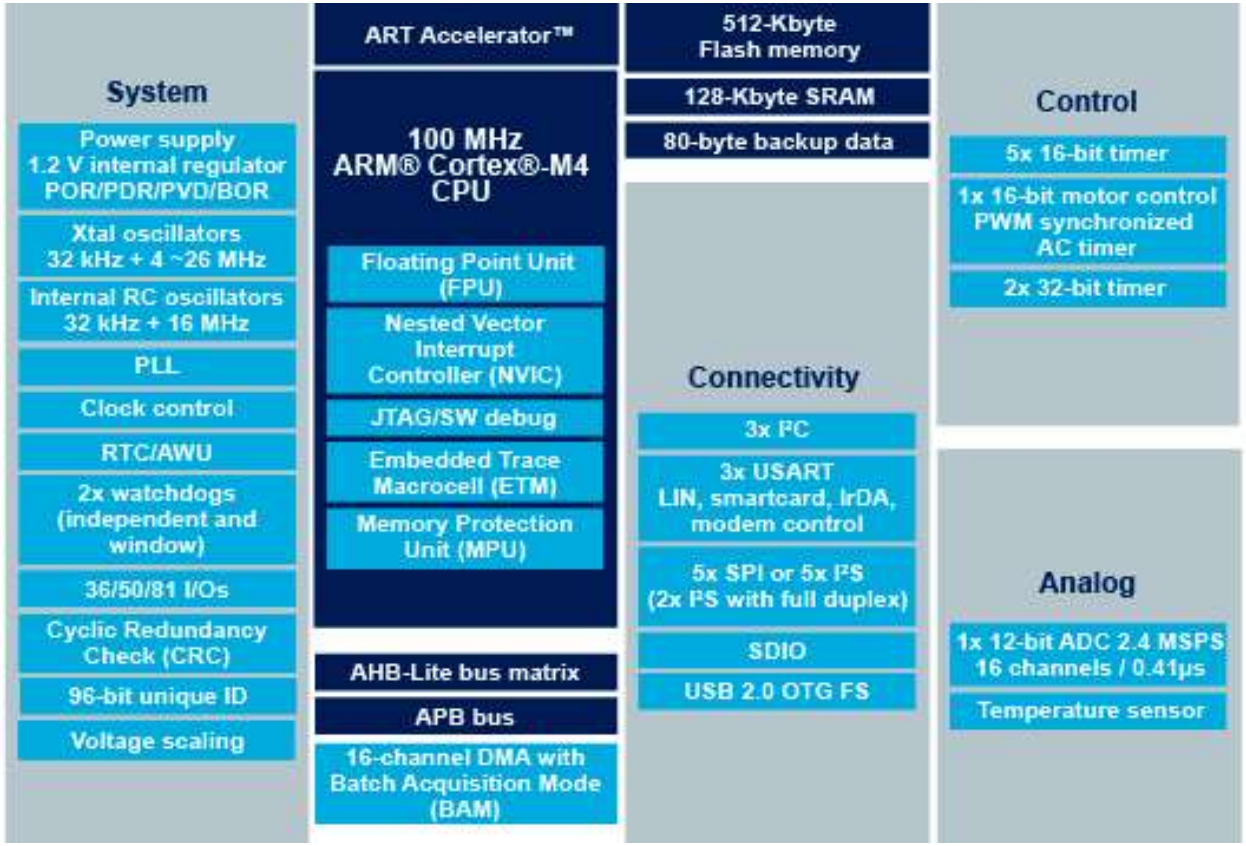


Development Tools

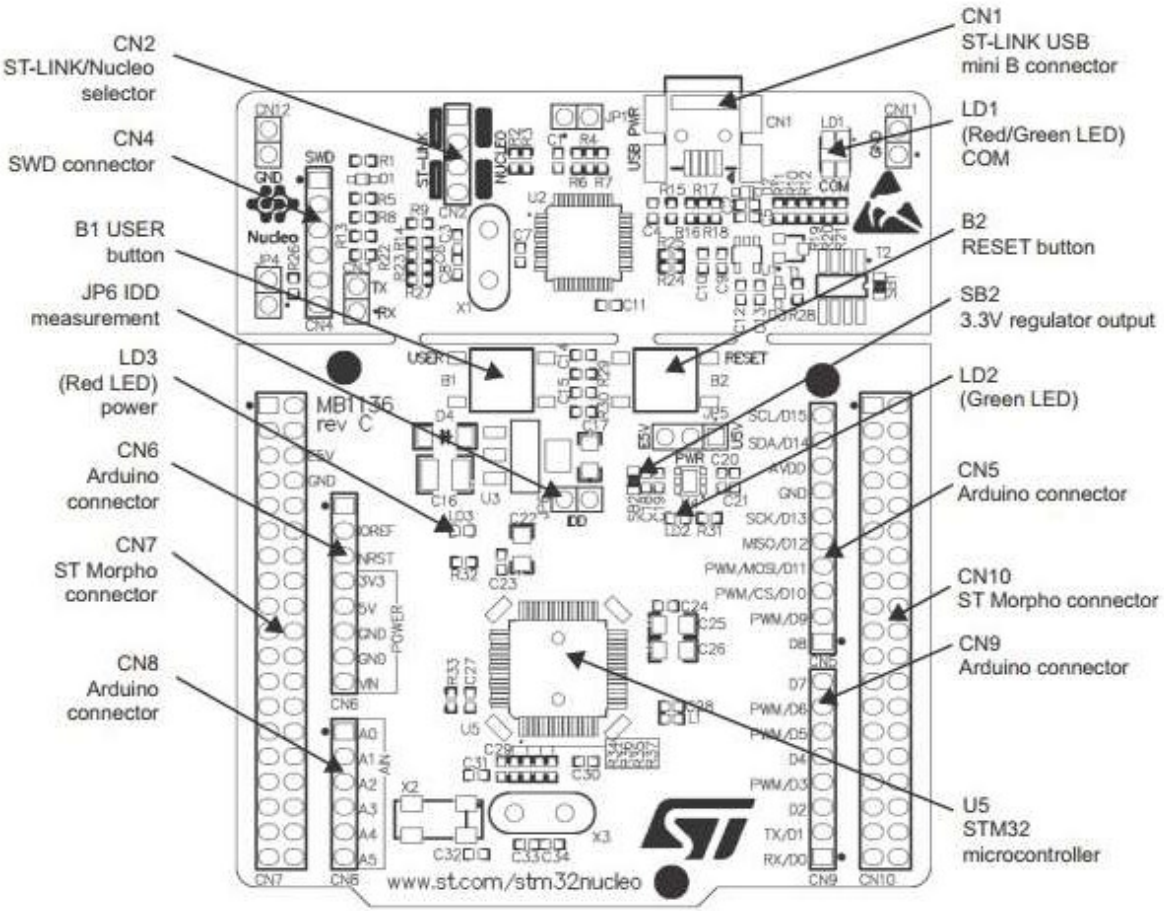
STM32CubeIDE

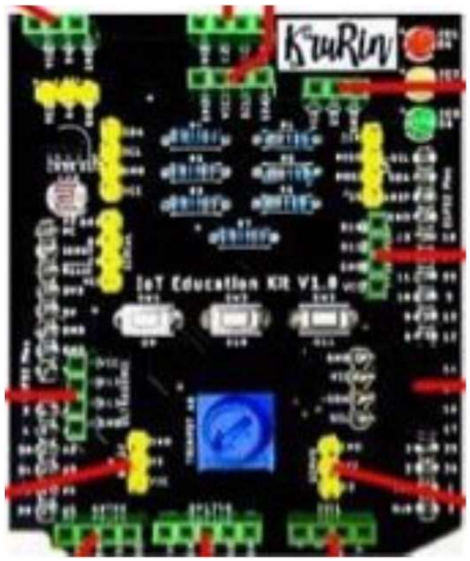
- STM32CubeIDE
  - an advanced C/C++ development platform with peripheral configuration, code generation, code compilation, and debug features for STM32 microcontrollers and microprocessors. It is based on the Eclipse®/CDT framework, GCC toolchain for the development and GDB for the debugging.
- With STM32CubeIDE, you can
  - select the appropriate STM32 device corresponding to your needs
  - configure the device using STM32CubeMX
  - develop and debug applications on top of Arm® Cortex®-M

STM32F411RE Components

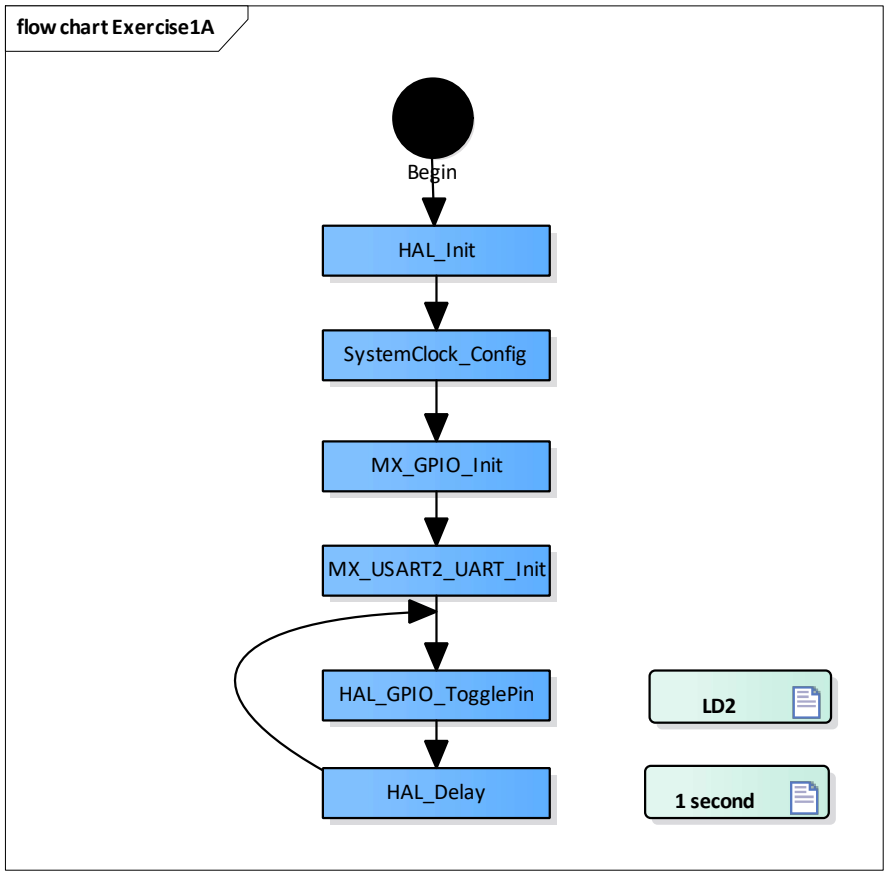


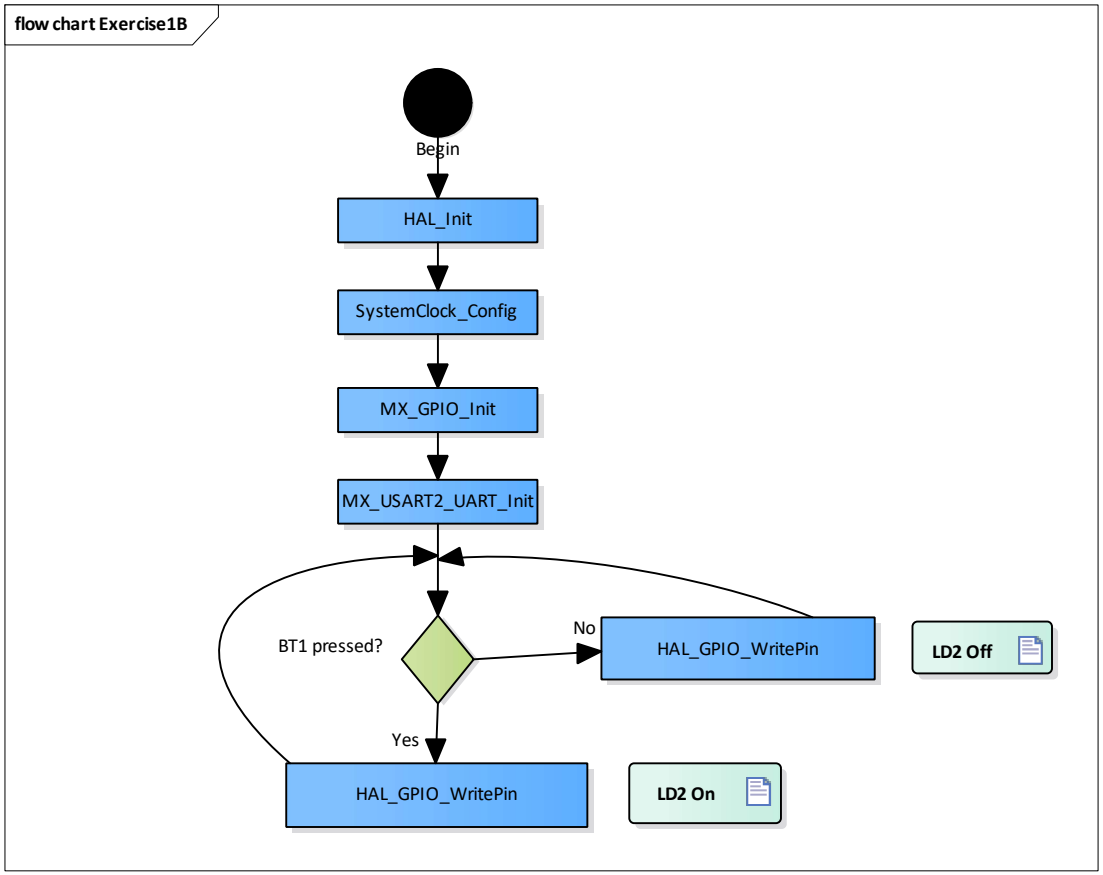
Boards


CN2
ST-LINK/Nucleo
selector
CN4
SWD connector
B1 USER
button
JP6 IDD
measurement
LD3
(Red LED)
power
CN6
Arduino
connector
CN7
ST Morpho
connector
CN8
Arduino
connector
CN1
ST-LINK USB
mini B connector
LD1
(Red/Green LED)
COM
B2
RESET button
SB2
3.3V regulator output
LD2
(Green LED)
CN5
Arduino connector
CN10
ST Morpho connector
CN9
Arduino connector
U5
STM32
microcontroller
MB1136
rev C
www.st.com/stm32nucleo


KuRiv
IoT Education Kit V1.0

Exercise1: It’s time to start new learning





Some Hints:

- see STM32F4 HAL and low-layer drivers User manual
  - HAL_Delay see page 48
  - HAL_GPIO_TogglePin see page 415
  - HAL_GPIO_ReadPin see page 414
  - HAL_GPIO_WritePin see page 414
- See details of Ports and Pins in main.h, also lines 200 and 206 in the original code for sample usage
- Example code for related HAL functions

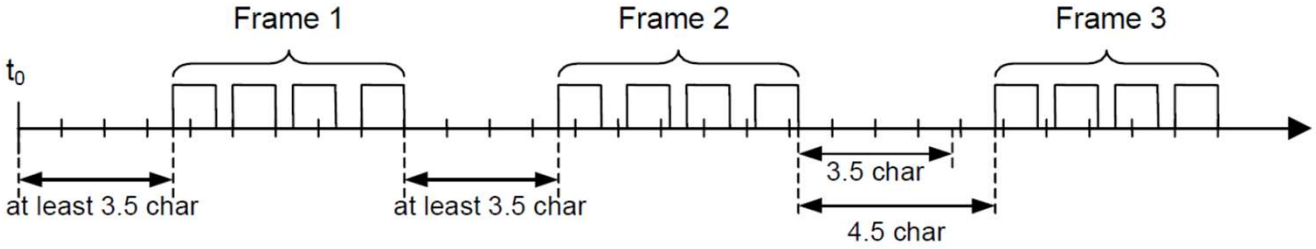
```
HAL_Delay(5000);
HAL_GPIO_TogglePin (RED_GPIO_Port, RED_Pin);
HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_7);
HAL_GPIO_WritePin(RED_GPIO_Port, RED_Pin, SET);
```

Modbus End of Frame Detection

2.5.1.1 MODBUS Message RTU Framing

A MODBUS message is placed by the transmitting device into a frame that has a known beginning and ending point. This allows devices that receive a new frame to begin at the start of the message, and to know when the message is completed. Partial messages must be detected and errors must be set as a result.

In RTU mode, message frames are separated by a silent interval of at least 3.5 character times. In the following sections, this time interval is called t3,5.



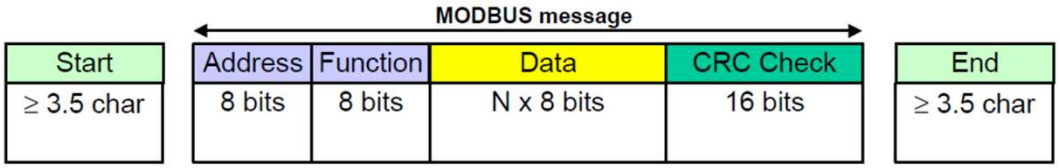
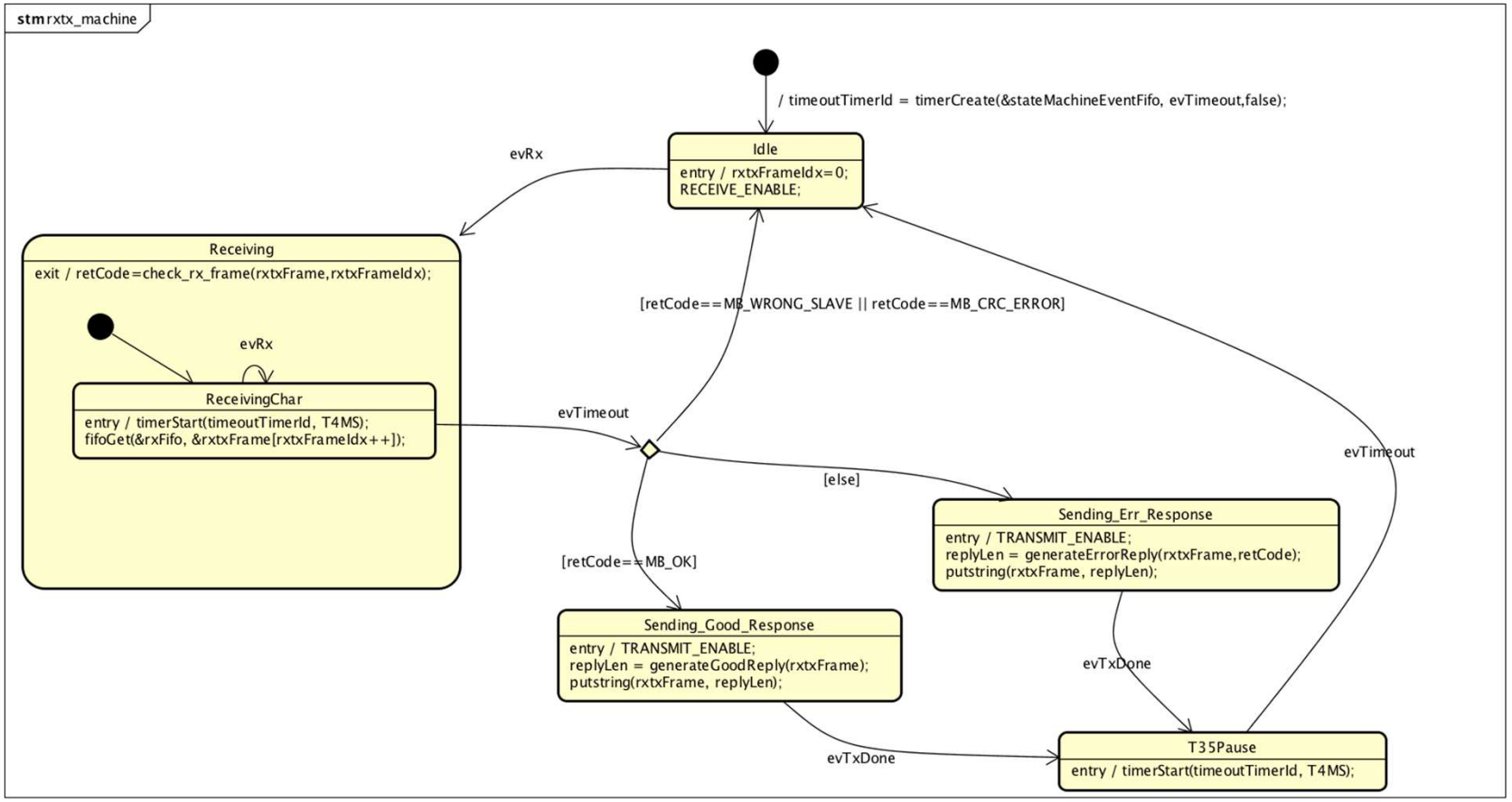
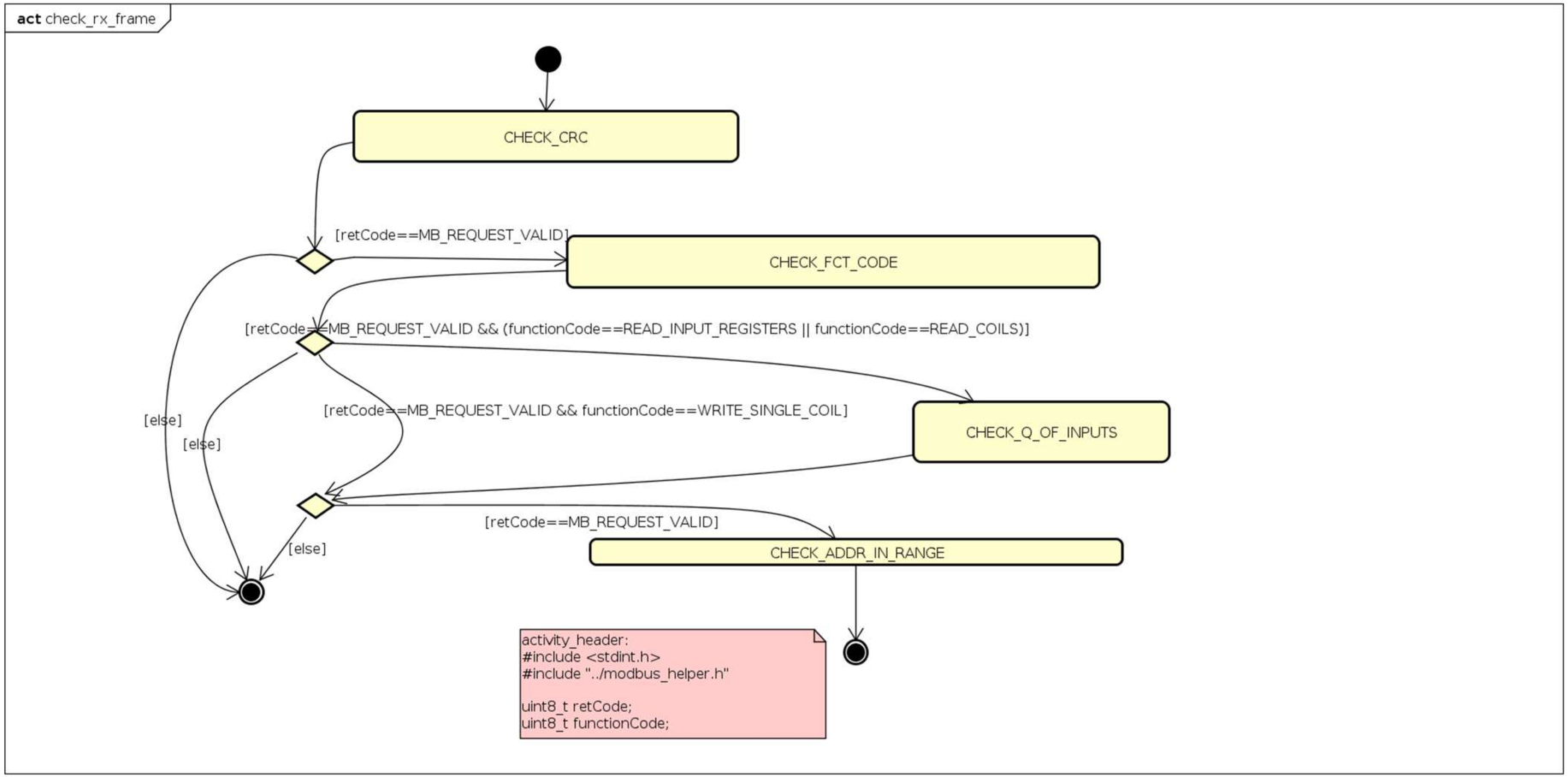


Figure 13: RTU Message Frame

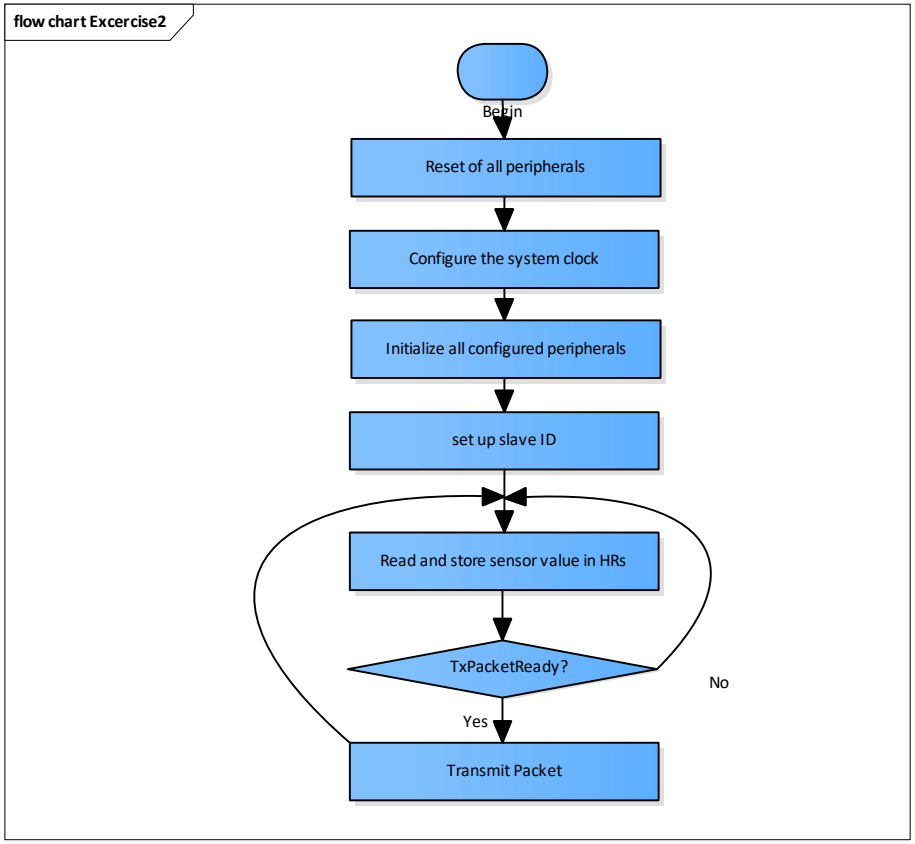
Modbus Frame Reception



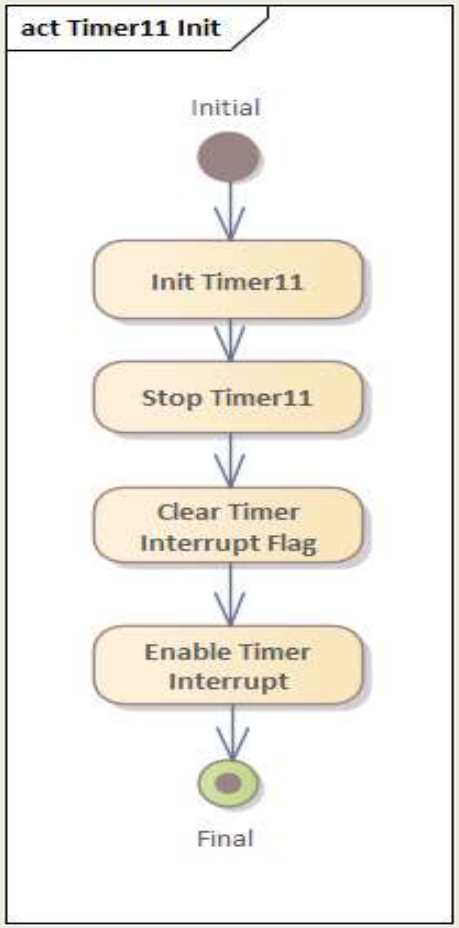
Function Code (FC) Handler

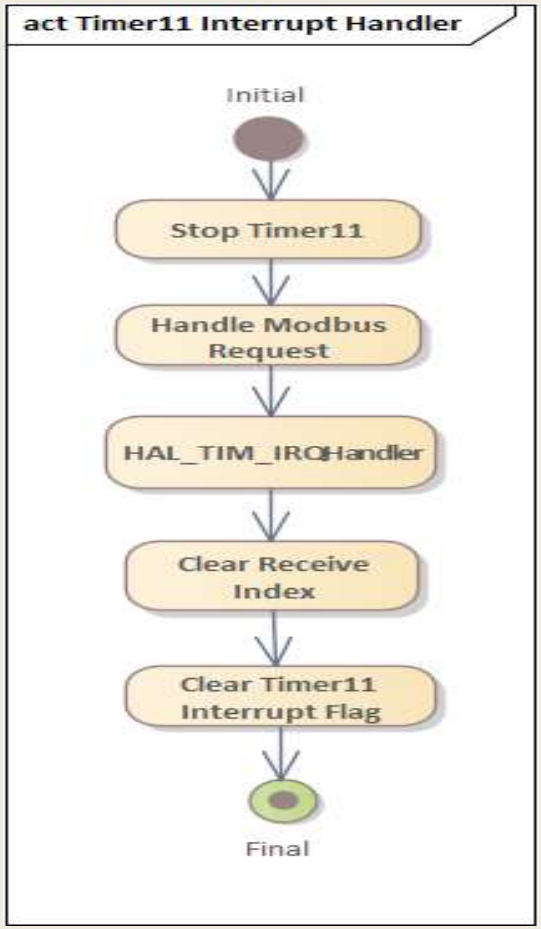


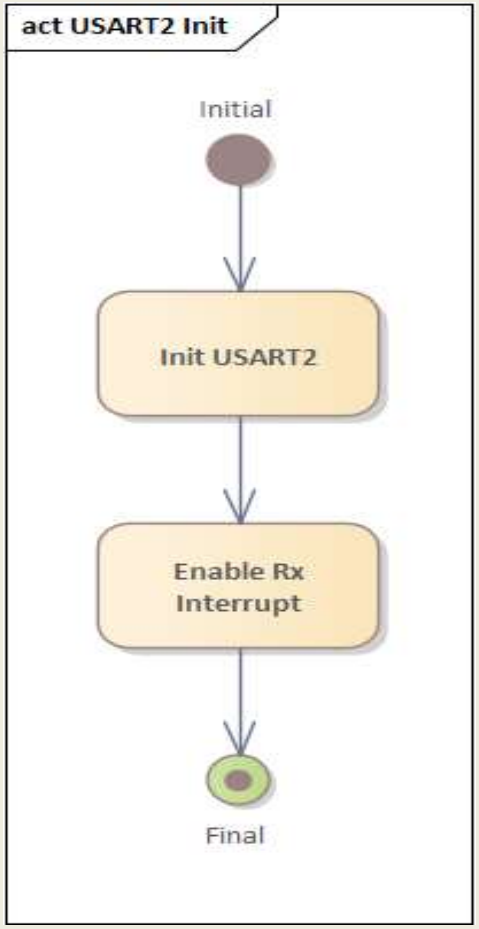
Main flowchart

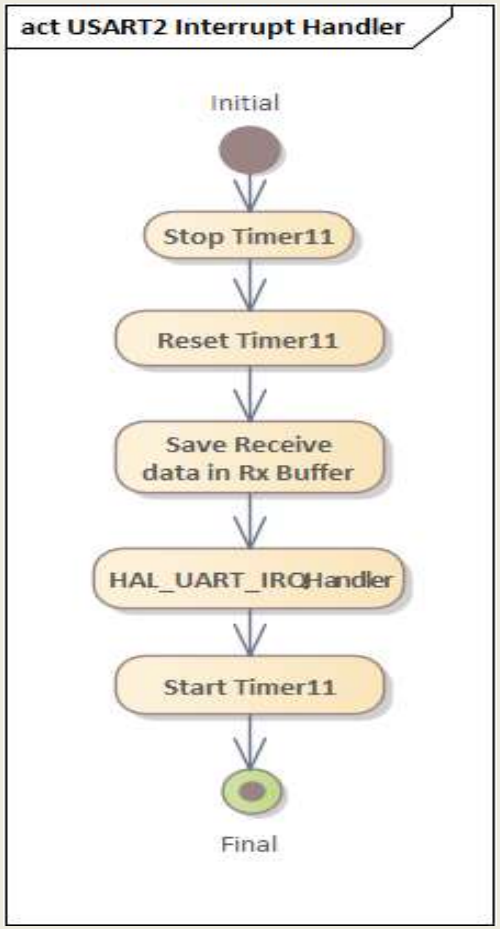


Activity Diagrams









Excercise2: It's time to write your own low level code

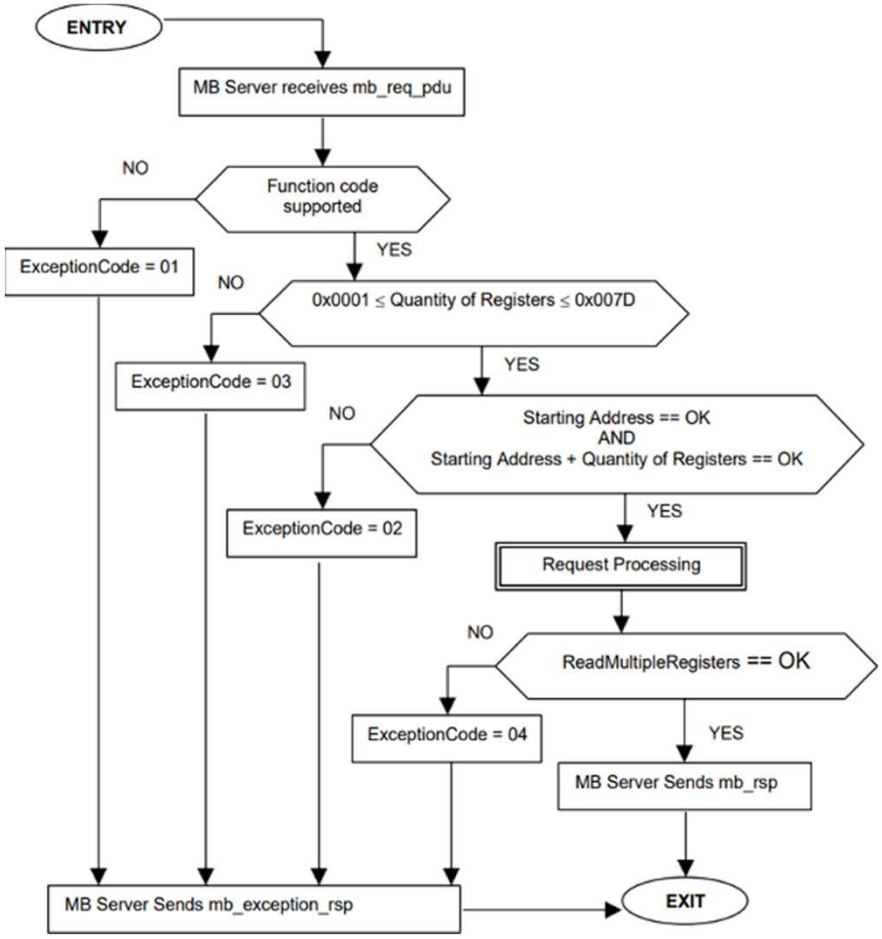
- #1 263 main.c
- #2 300 main.c
- #3 237 stm32f4xx_it.c
- #4 239-241 stm32f4xx_it.c
- #5 253 stm32f4xx_it.c

Provided answers

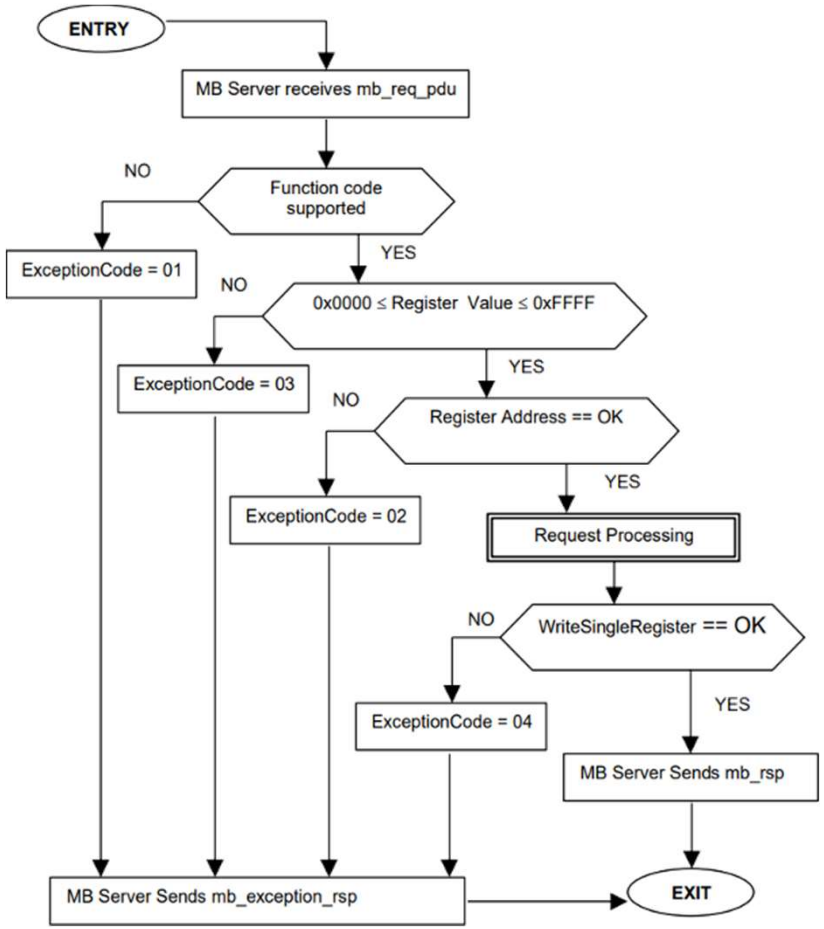
```
TIM11->CNT = 0xffff;
USART2->CR1 = USART2->CR1 | 0x20;
TIM11->CR1 |= 0x01;
USART2->CR1 = USART2->CR1 & 0x20;
TIM11->DIER |= 1;
TIM11->CR1 &= 0x01;
TIM11->DIER &= 1;
TIM11->CNT = 0;
```

Ref. Manual (C.7)
USART starts at p. 505
Timer11 starts at p. 375

FC 03 (see https://ozeki.hu/p_5878-mobdbus-function-code-3-read-multiple-holding-registers.html)



FC 06 (see https://ozeki.hu/p_5881-mobdbus-function-code-6-write-single-holding-register.html)



Exercise3: It's time to write your own Modbus Slave

- Import Exercise3.zip to STM32CubeIDE
- Study FC 03 implementation from Modbus.c
- Implement FC 06 in Modbus.c, try to reuse the code from FC 03
- Add three HRs, HR[5] to HR[7], to control three LEDs on the shield
- Test your FC 06 using qModMaster
- Send your testing clip to the instructor via an email

Dependable Modbus Slave

- Your Modbus slave is expected to serve 24 hours/day
- Environment would cause your Modbus slave to hang
- We need a mechanism to ensure that your Modbus slave is always responsive
- Watchdog timer will reset your Modbus slave if it is hung

Exercise4: It’s time to make your Modbus Slave dependable

- Study STM32 Watchdog timer from the reference manual
- Add Watchdog timer to your FC 06 Modbus slave
- Implement and test your Watchdog implementation
- Send your testing screen capture to the instructor via an email