**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KĨ THUẬT MÁY TÍNH**



**BÁO CÁO QUÁ TRÌNH**

**THỰC TẬP DOANH NGHIỆP**

1. Yêu cầu:

* Tổng hợp tài liệu cần thiết
* Cài đặt phần mềm
* Khởi tạo project điều khiển (bật/tắt) LED trên Kit Artery AT32F403A

1. Triển khai(registers):
2. Tổng hợp các tài liệu cần thiết

Tất cả các tài liệu bao gồm Reference Manual, Technical Manual, User Manual, chip Datasheet và Schematic được tổng hợp tại link sau:

[Github\_LEDToggle\_ReferenceDocuments](https://github.com/hniman135/LED_toggle-register-/tree/main/Reference%20Documents)

1. Cài đặt phần mềm

Các phần mềm được đề xuất và Add-ons cần thiết

A screenshot of a computer

Description automatically generated

* Keil C v5

A screenshot of a computer

Description automatically generated

\*Add-on: [Keil 5 Add-on](https://www.arterychip.com/download/PACK/Keil5_AT32MCU_AddOn_V2.2.6.zip)

[Keil 4 Add-on](https://www.arterychip.com/download/PACK/Keil4_AT32MCU_AddOn_V2.2.3.zip)

* IAR

\*Add-on: [IAR Add-on](https://www.arterychip.com/download/PACK/IAR_AT32MCU_AddOn_V2.1.8.zip)

* AT32 IDE (currently use)

\*SDK Install:

A screenshot of a computer

Description automatically generated

1. Khởi tạo project điều khiển (bật/tắt) LED trên Kit Artery AT32F403A

* Include, definition và typedef

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

* GPIO Configuration

A screen shot of a computer code

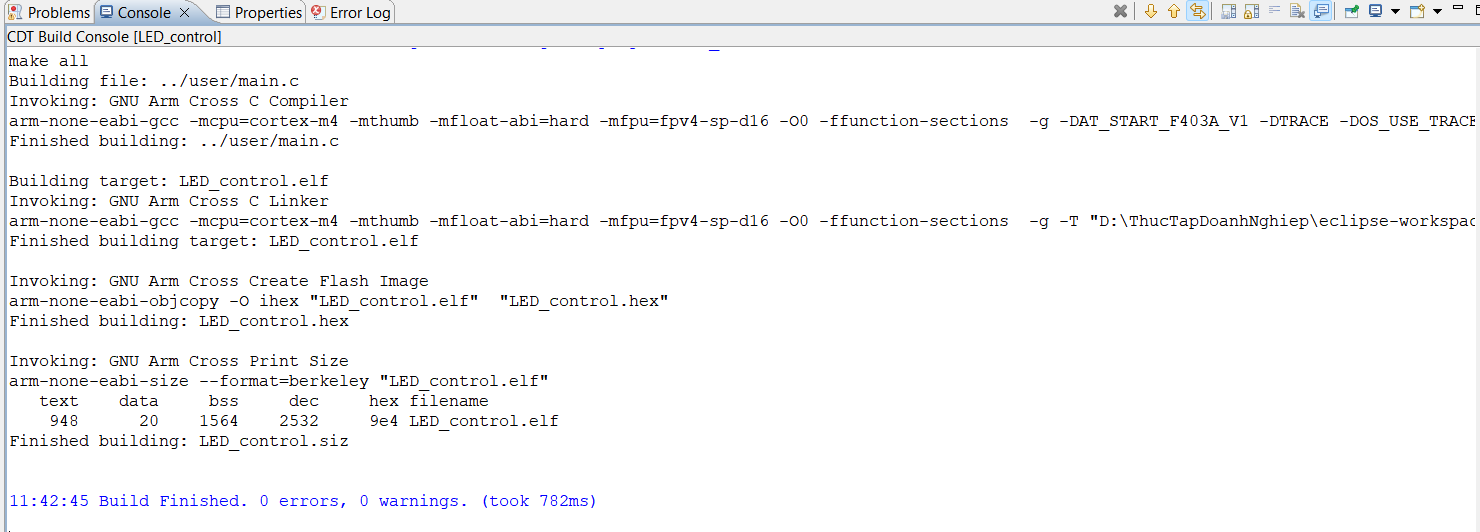
Description automatically generated

* Main code

A computer screen shot of a computer code

Description automatically generated

* Build result

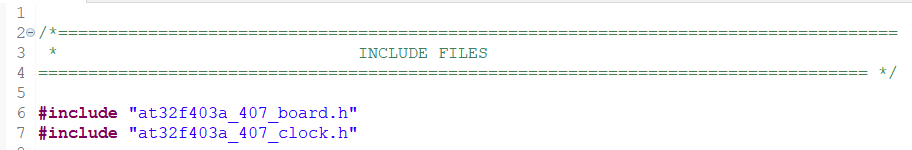


1. Project link & Demo video

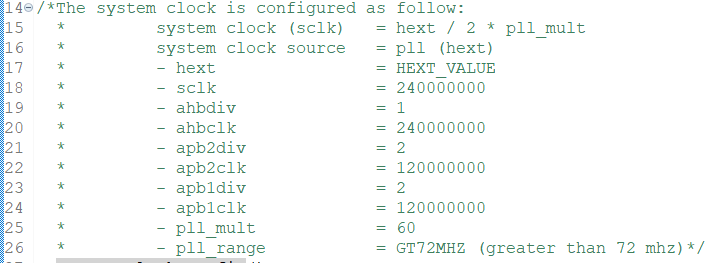
* Github: [AT32F403A\_LEDControl](https://github.com/hniman135/AT32F403A_LEDControl)
* Demo: [Demo\_video](https://drive.google.com/file/d/1nOafXAwdr_Ka8oCai4fVoG4V3Zo0DHgb/view?usp=sharing)

1. Triển khai (library)

* Include:



* system\_clock\_config():



* + system clock source (= 240 000 000) = pll (hext): clock ngoại thạch anh (HEXT\_VALUE = 8 000 000) \* pll\_mult(=60)
  + => PLL\_range greater than 72mhz
  + apb2clk ( = sclk / apb2div(=2) ): 120 000 000 (maximum frequency of APB1/APB2 )

**void** **system\_core\_clock\_update**(**void**)

{

uint32\_t hext\_prediv = 0, pll\_mult = 0, pll\_mult\_h = 0, pll\_clock\_source = 0, temp = 0, div\_value = 0;

crm\_sclk\_type sclk\_source;

**static** **const** uint8\_t sys\_ahb\_div\_table[16] = {0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 3, 4, 6, 7, 8, 9};

/\* get sclk source \*/

sclk\_source = crm\_sysclk\_switch\_status\_get();

**switch**(sclk\_source)

{

**case** *CRM\_SCLK\_HICK*:

**if**(((CRM->misc3\_bit.hick\_to\_sclk) != *RESET*) && ((CRM->misc1\_bit.hickdiv) != *RESET*))

system\_core\_clock = HICK\_VALUE \* 6;

**else**

system\_core\_clock = HICK\_VALUE;

**break**;

**case** *CRM\_SCLK\_HEXT*:

system\_core\_clock = HEXT\_VALUE;

**break**;

**case** *CRM\_SCLK\_PLL*:

pll\_clock\_source = CRM->cfg\_bit.pllrcs;

{

/\* get multiplication factor \*/

pll\_mult = CRM->cfg\_bit.pllmult\_l;

pll\_mult\_h = CRM->cfg\_bit.pllmult\_h;

/\* process high bits \*/

**if**((pll\_mult\_h != 0U) || (pll\_mult == 15U)){

pll\_mult += ((16U \* pll\_mult\_h) + 1U);

}

**else**

{

pll\_mult += 2U;

}

**if** (pll\_clock\_source == 0x00)

{

/\* hick divided by 2 selected as pll clock entry \*/

system\_core\_clock = (HICK\_VALUE >> 1) \* pll\_mult;

}

**else**

{

/\* hext selected as pll clock entry \*/

**if** (CRM->cfg\_bit.pllhextdiv != *RESET*)

{

hext\_prediv = CRM->misc3\_bit.hextdiv;

/\* hext clock divided by 2 \*/

system\_core\_clock = (HEXT\_VALUE / (hext\_prediv + 2)) \* pll\_mult;

}

**else**

{

system\_core\_clock = HEXT\_VALUE \* pll\_mult;

}

}

}

**break**;

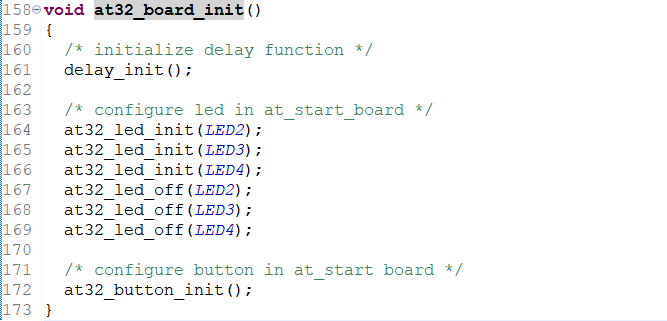
**default**:

system\_core\_clock = HICK\_VALUE;

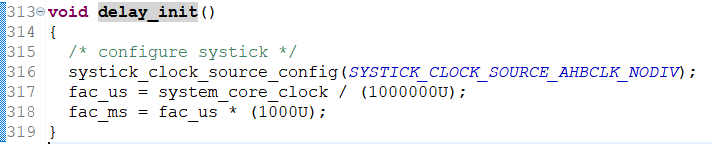
**break**;

}

* at32\_board\_init():



* + delay\_init()



* + - Enable ahbclk
  + At32\_led\_init()

A screenshot of a computer code

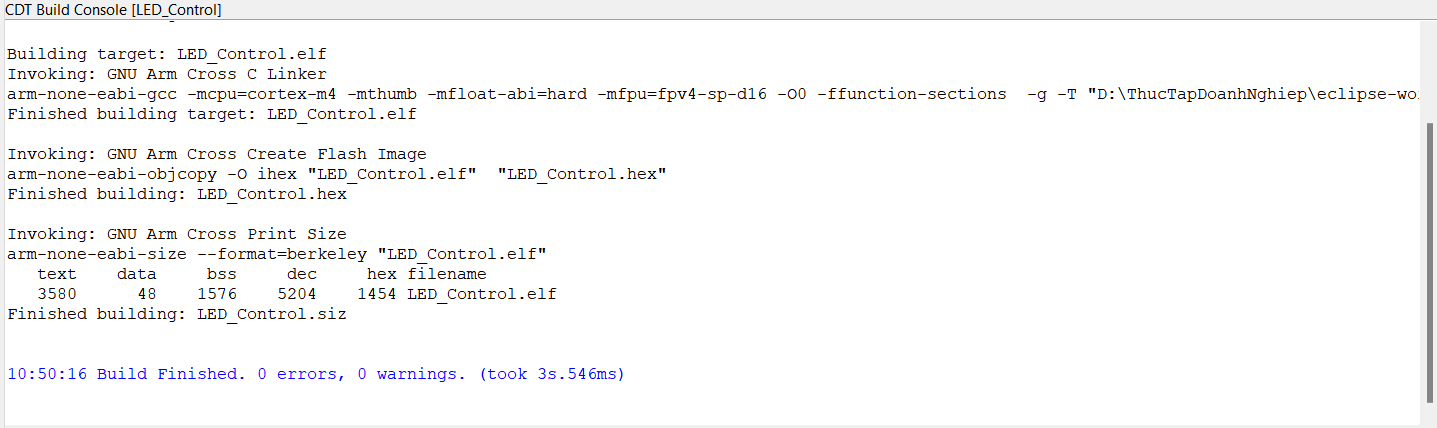
Description automatically generated

* + - Enable CRM\_GPIOD\_PERIPH\_CLOCK
    - Config mode output push-pull
  + At32\_led\_toggle()

A computer code with black text

Description automatically generated with medium confidence

* + - Toggle trạng thái của LED (ODT register)
* Build result:



* Project link & Demo video:

Github: [AT32F403A\_LEDControlLibrary](https://github.com/hniman135/LEDControlLibrary)

Demo: [Demo Video](https://drive.google.com/file/d/1Twy4f6-thkCt3jDtzZjDR9ezOerJaH-c/view?usp=sharing)