**EE 354**
**Assignment 5**
**ARM Introduction**

**October 22, 2018**
**Due: October 29, 2018**

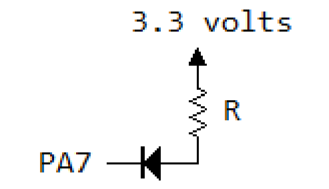
Purchase and ARM NUCLEO-F446RE from Mouser at
http://www.mouser.com/ProductDetail/STMicroelectronics/NUCLEO-F446RE/?qs=sGAEpiMZZMvc81WFyF5EdsQmNbgZtBbGR8SlOiVkIXM%3d
or get one from the EE office in KC 266.

Read the notes at https://csserver.evansville.edu/~blandfor/EE354/NucleoBoardUserManual.pdf
and get the program STM32F446Template running on your board:
https://csserver.evansville.edu/~blandfor/EE354/STM32F446Template.zip

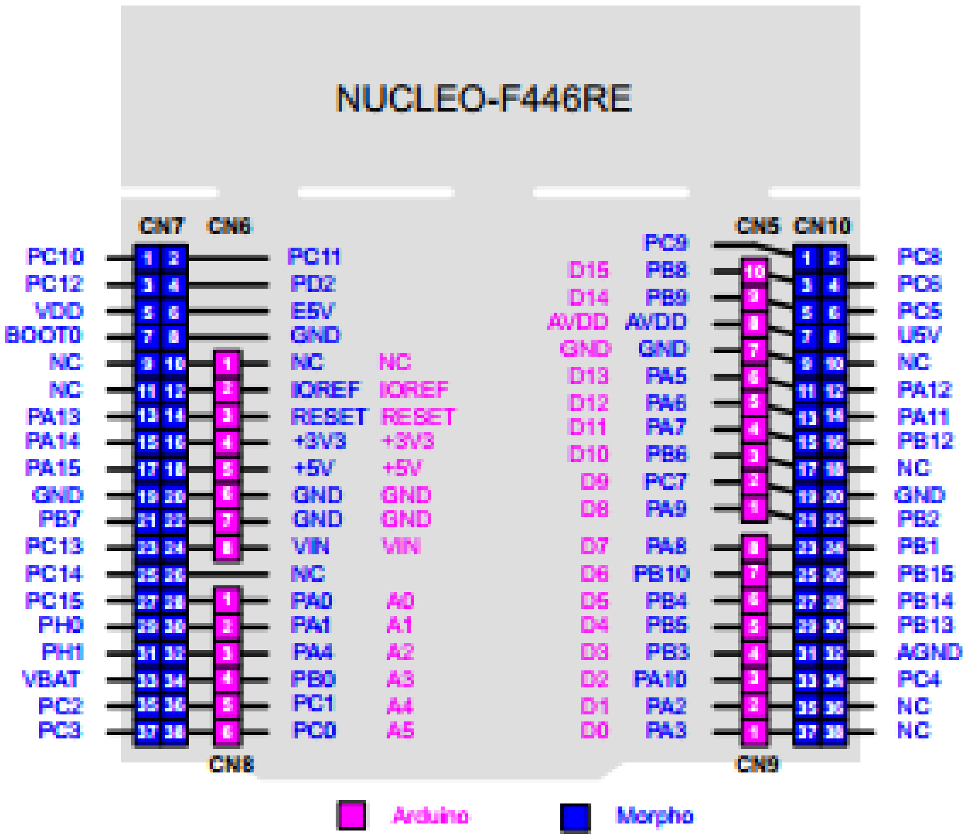
Once you get the template loading and running correctly, create a copy of the template and name it LEDPA7. Change the c file named KeilTempleV5.c to the program below and load this program on you board. Connect an LED to PA7. Use an oscilloscope to verify that PA7 oscillates at about 6 MHz. Uncomment the delay loop on line 21 and change it to:

```
   for(i=0;i<800000;i++);
```

This will allow PA7 to oscillate at about 8 times a second. Connect an LED to PA7 and bring your board to class to show that it works.

| | |
|---|---|
| Set R = 470 Ω |  |

```
#include "stm32f446.h"
/*stm446Template.c              July 1, 2017
  This program toggles a bit on PA7 as fast as possible
*/
void InitializeClock(void);
int main()
  {int i, tmp;
   //Clock bits
   InitializeClock();
   RCC_AHB1ENR |= 1;         //Bit 0 is GPIOA clock enable bit
   //I/O bits
   GPIOA_MODER |= 0x4000;    //Bits 15-14 = 01 for digital output on PA7
   //OTYPER register resets to 0 so it is push/pull by default
   GPIOA_OSPEEDER |= 0xC000;  //Bits 15-14 = 11 for high speed on PA7
   //PUPDR defaults to no pull up no pull down
   //Main program loop
   tmp = 0;
   while(1)
     {GPIOA_ODR = tmp;       //Only PA7 is set up for output so other bits
      tmp = ~tmp;            //  have no effect.
      for(i=0;i<800000;i++);
     }
  }
//This function resets the system clock to 168 MHz.
  void InitializeClock()
      {RCC_CFGR = 0x00000000;         //Reset Clock Configuration Register
       RCC_CR &= 0xFEF6FFFF;          //Reset HSEON, CSSON and PLLON Bits
       RCC_CR |= (1 << 16);           //Turn on HSE clock
       while((RCC_CR & (1 << 17)) == 0); //Wait until HSE is ready
       RCC_CR |= (1 << 19);
       RCC_PLLCFGR = 0x27405A08;      //Set PLLP = 0, PLLN = 360, PLLM = 8,
                                      //PLLQ = 7, PLL Src = HSE
       RCC_CR |= (1 << 24);           //Enable PLL on
       while((RCC_CR & (1 << 25)) == 0); //Wait for PLL to lock on
       RCC_CFGR = 0x9402;             // APB2/2, APB1/4, AHB/1
       FLASH_ACR &= 0xFFFFFFF8;       //Set flash wait states to 5
       FLASH_ACR |= 0x5;
      }
```


NUCLEO-F446RE
CN7
CN6
CN5
CN10
PC10 1 2 PC11
PC12 3 4 PD2
VDD 5 6 E5V
BOOT0 7 8 GND
NC 9 10 1 NC NC
NC 11 12 2 IOREF IOREF
PA13 13 14 3 RESET RESET
PA14 15 16 4 +3V3 +3V3
PA15 17 18 5 +5V +5V
GND 19 20 6 GND GND
PB7 21 22 7 GND GND
PC13 23 24 8 VIN VIN
PC14 25 26 NC
PC15 27 28 1 PA0 A0
PH0 29 30 2 PA1 A1
PH1 31 32 3 PA4 A2
VBAT 33 34 4 PB0 A3
PC2 35 36 5 PC1 A4
PC3 37 38 6 PC0 A5
CN8
PC9 1 2 PC8
D15 PB8 10 3 4 PC6
D14 PB9 9 5 6 PC5
AVDD AVDD 8 7 8 U5V
GND GND 7 9 10 NC
D13 PA5 6 11 12 PA12
D12 PA6 5 13 14 PA11
D11 PA7 4 15 16 PB12
D10 PB6 3 17 18 NC
D9 PC7 2 19 20 GND
D8 PA9 1 21 22 PB2
D7 PA8 8 23 24 PB1
D6 PB10 7 25 26 PB15
D5 PB4 6 27 28 PB14
D4 PB5 5 29 30 PB13
D3 PB3 4 31 32 AGND
D2 PA10 3 33 34 PC4
D1 PA2 2 35 36 NC
D0 PA3 1 37 38 NC
CN9
Arduino
Morpho