BUKU PANDUAN PRAKTIKUM (DARING)

MIKROKONTROLLER

LABORATORIUM TEKNIK ELEKTRO

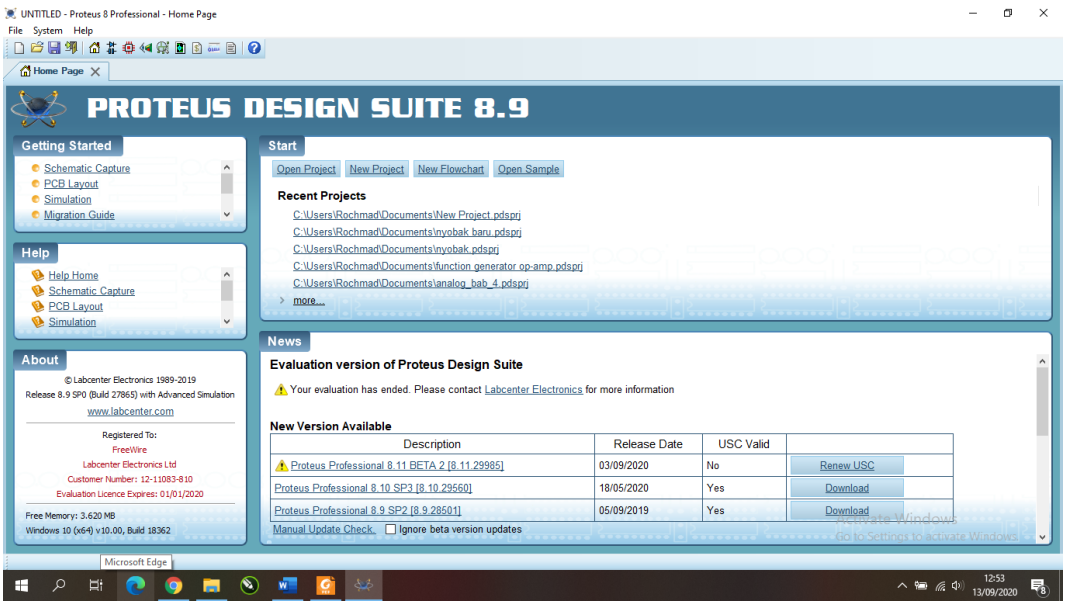
UNIVERSITAS MUHAMMADIYAH MALANG

Laboratorium Teknik Elektro
Universitas Muhammadiyah Malang

PENDAHULUAN

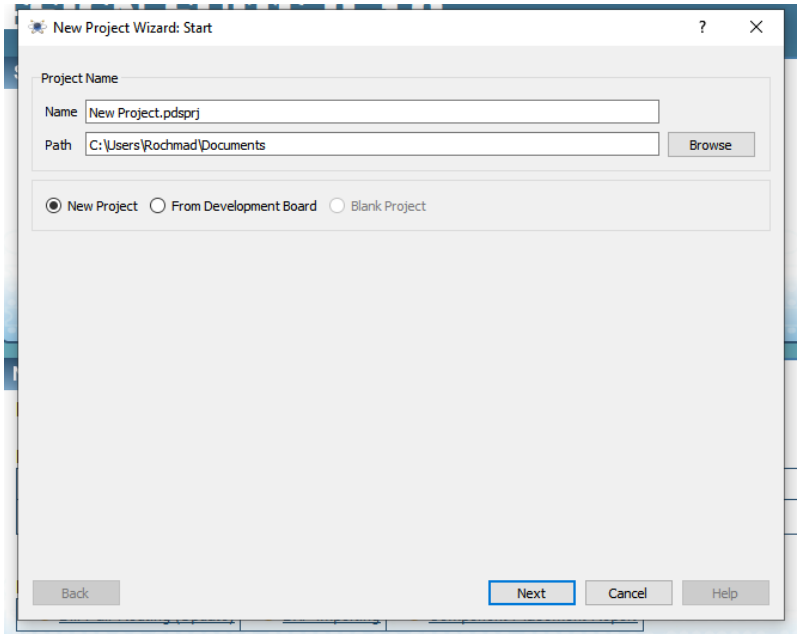
Proteus professional merupakan suatu software yang digunakan untuk melakukan simulasi untuk perangkat elektronik oleh para penggiat atau develop, mulai dari rangkaian yang paling sederhana hingga rangkaian yang sangat kompleks. Dengan adanya software ini dapat memudahkan bagi para desainer dalam melakukan simulasi rangkaian elektronik dengan desain yang telah dirancang dan sangat membantu sekali dikarenakan dana mengurangi kesalahan yang tidak diinginkan. Software ini memiliki banyak kelebihan salah satunya yaitu mode simulasi yang pada software ini tampilkan yaitu paket ISIS dimana terdapat banyak sekali komponen-komponen elektronika baik komponen aktif maupun pasif. Selain itu juga terdapat beberapa alat ukur seperti Voltmeter DC/ac, Amperemeter DC/ac, osiloskop, function generator, dll. Dengan banyaknya kelebihan pada paket ISIS sangat cocok digunakan untuk mendesain suatu sistem yang diinginkan dan dapat mengurangi kesalahan yang tidak diinginkan sehingga menjadikan software ini menjadi salah satu software terbaik bagi para desainer khususnya dibidang elektronik.

Pada tampilan software proteus professional versi 8.9 dapat dilihat pada gambar berikut :



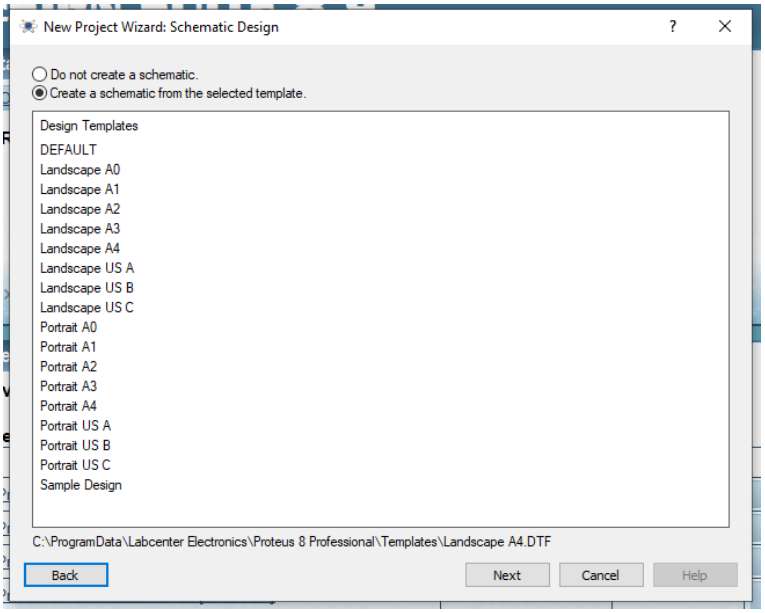
Gambar Tampilan Proteus Profesional 8.9

Proteus versi merupakan perbaikan dari versi sebelumnya dan tidak mengubah dari fungsinya sehingga tetap mudah dalam penggunaannya. Pada tampilan ini pengguna diharapkan untuk membuat projek terlebih dahulu dengan cara masuk menu File + New Project (CTRL + N) sehingga akan muncul Langkah-langkah sebagai berikut :



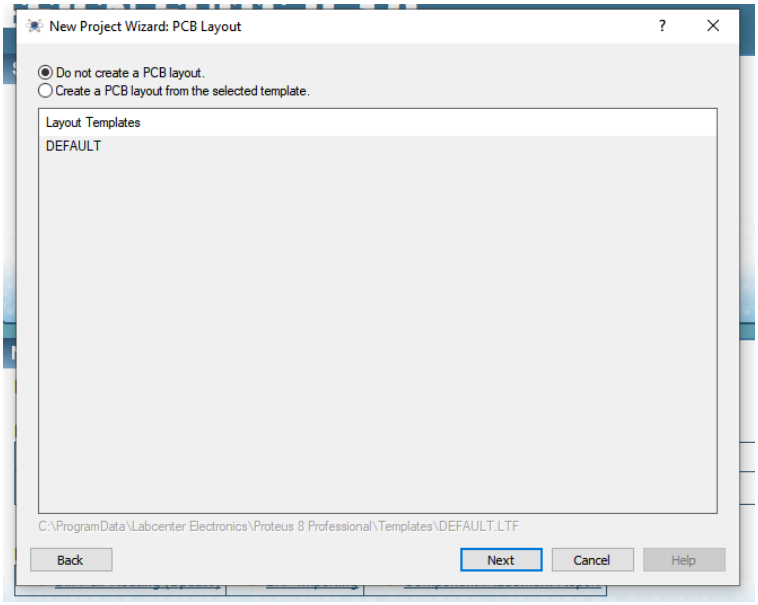
Gambar Langkah 1 New Project Wizard Start

Pada kotak dialog kali ini memberikan nama project yang diinginkan dan menentukan alamat direktori yang diinginkan untuk menyimpan projext yang telah dibuat.



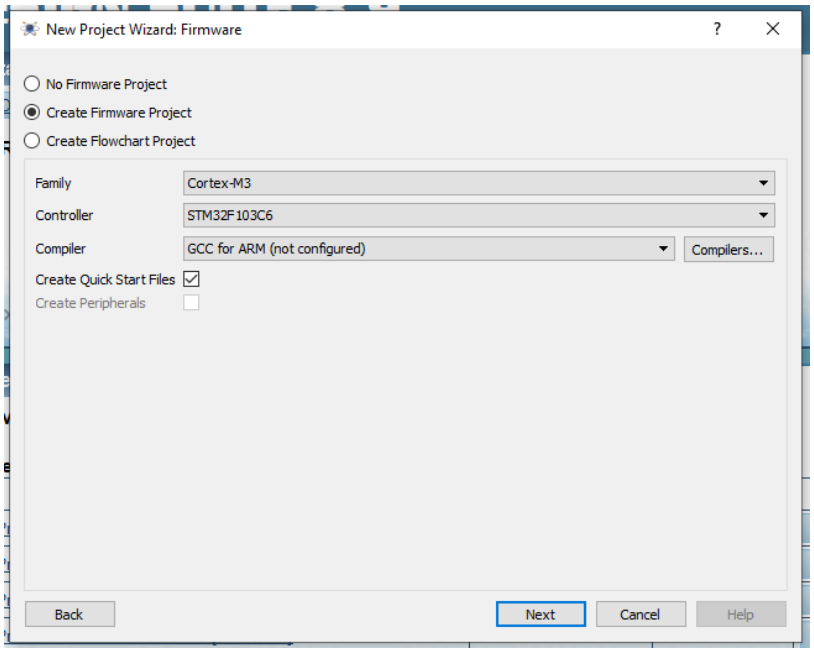
Gambar Langkah 2 New Project Wizard Schematic Design

Selanjutnya merupakan tampilan untuk menentukan ukuran template yang diinginkan pada project yang dibuat.



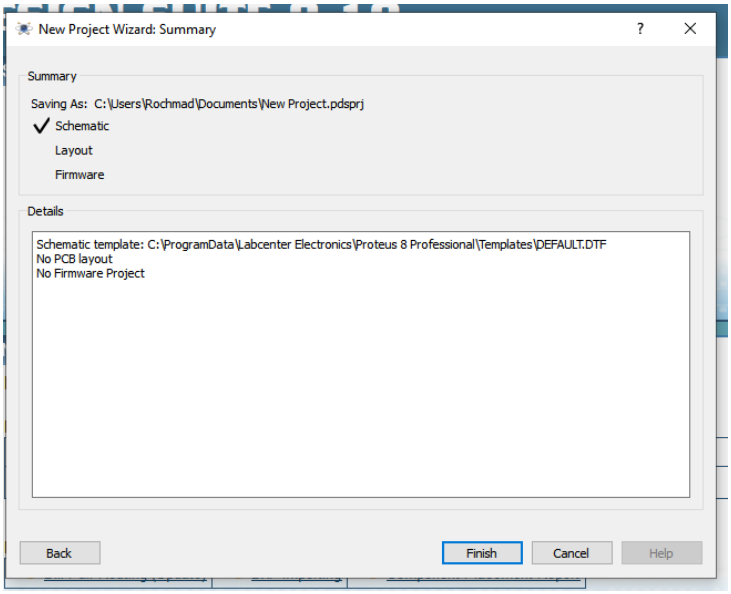
Gambar Langkah 3 New Project Wizard PCB Layout

Pada kotak dialog diatas merupakan untuk membuat PCB dari skematik yang telah dibuat pada project, sehingga pada praktikum ini dapat diabaikan dengan memilih “Do not create a PCB layout”.



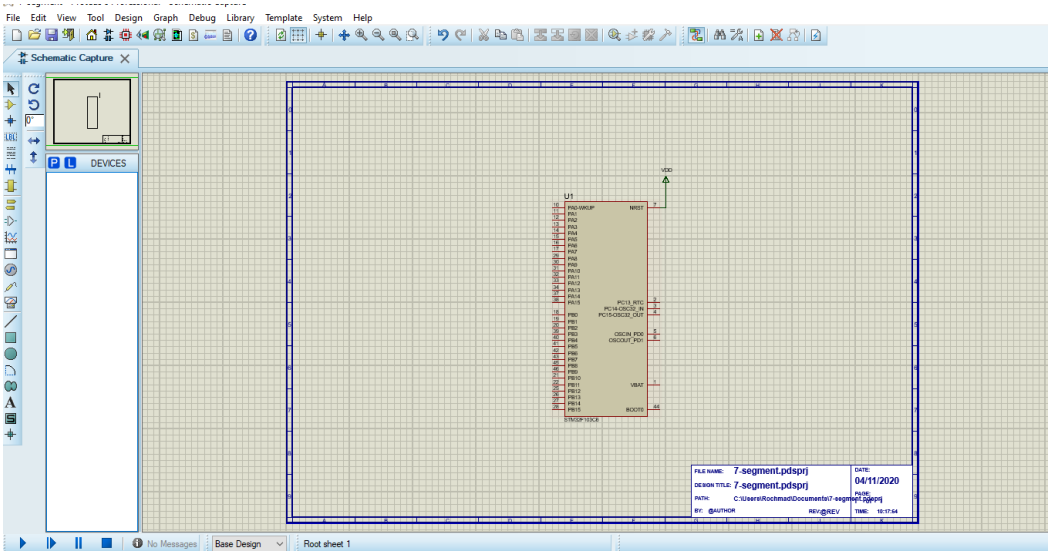
Gambar Langkah 4 New Project Wizard Firmware

Pada Langkah selanjutnya memilih firmware yang ingin dipakai atau memilih perangkat mikokontroller yang diinginkan, dimana praktikum ini menggunakan perangkat STM23F103C6 dengan Family Cortex-M3 sebagai mikrokontroller yang akan dipakai.



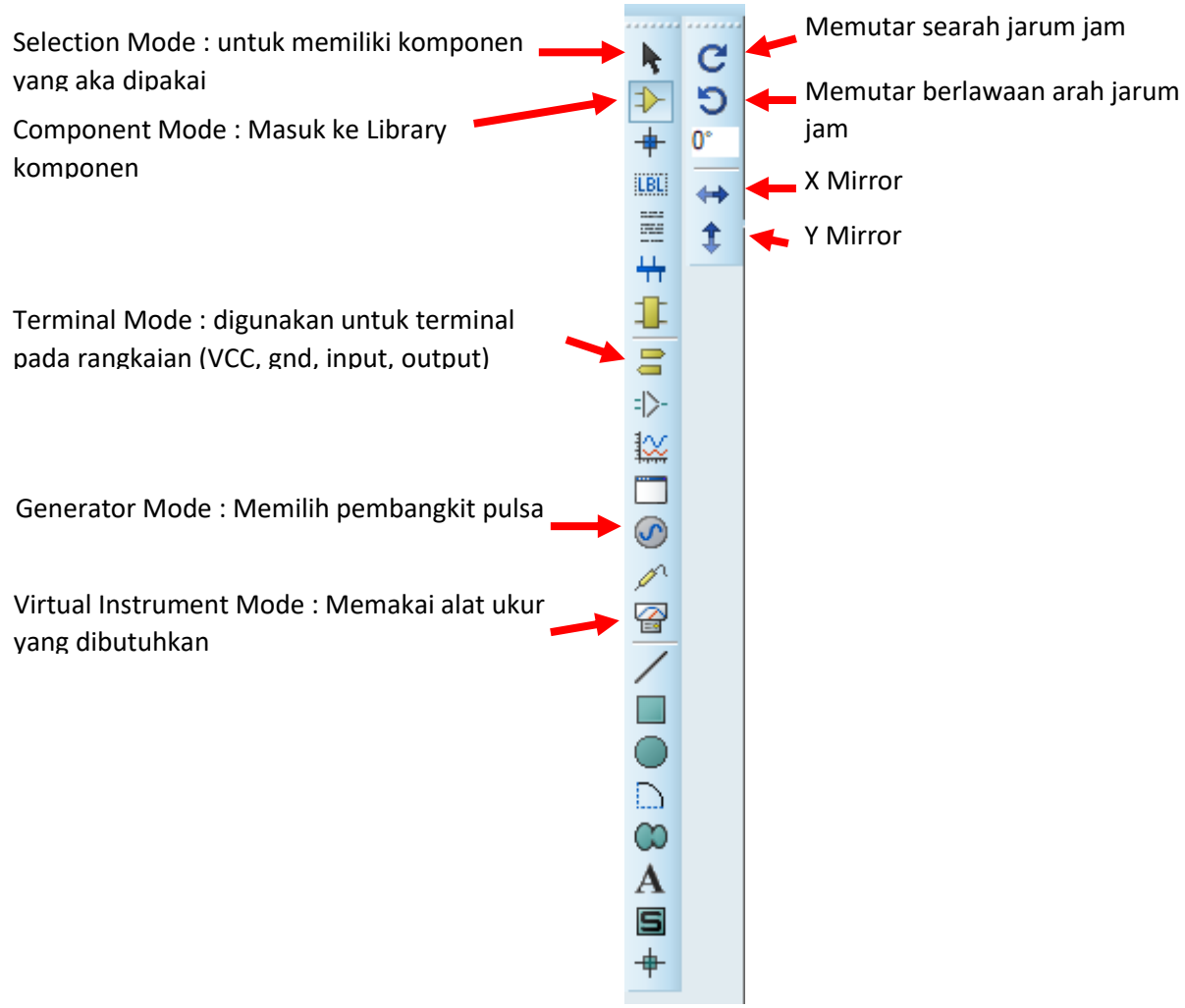
Gambar Langkah 5 New Project Wizard Summary

Pada proses selanjutnya maka akan muncul 2 centang yaitu pada “Schematic dan Firmware” lalu dapa diklik pada tombol finish untuk mengakhiri pembuatan project tersebut.

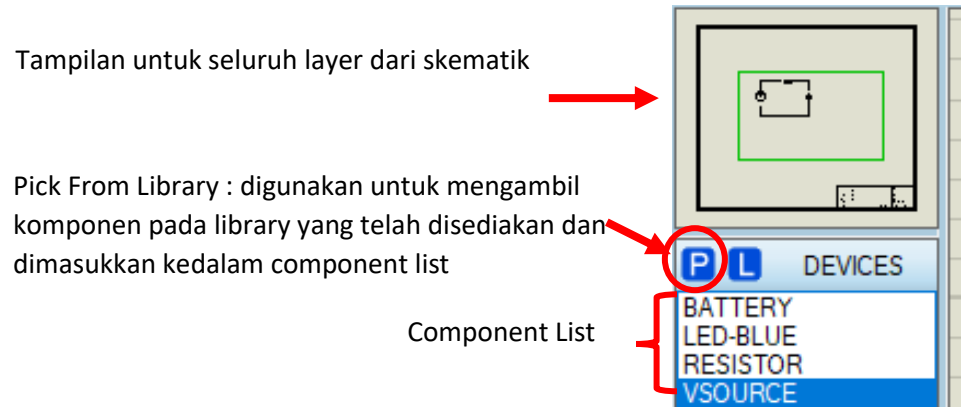


Gambar Tampilan Project Simulasi ISIS Proteus

Proses yang telah dibuat maka akan menghasilkan seperti tampilan pada gambar diatas. Sehingga dapat langsung dilakukan melakukan percobaan yang diinginkan.. Setelah pembuatan project selesai maka yang perlu diperhatikan kegunaan pada menu bar yang dijelaskan pada gambar dibawah ini :

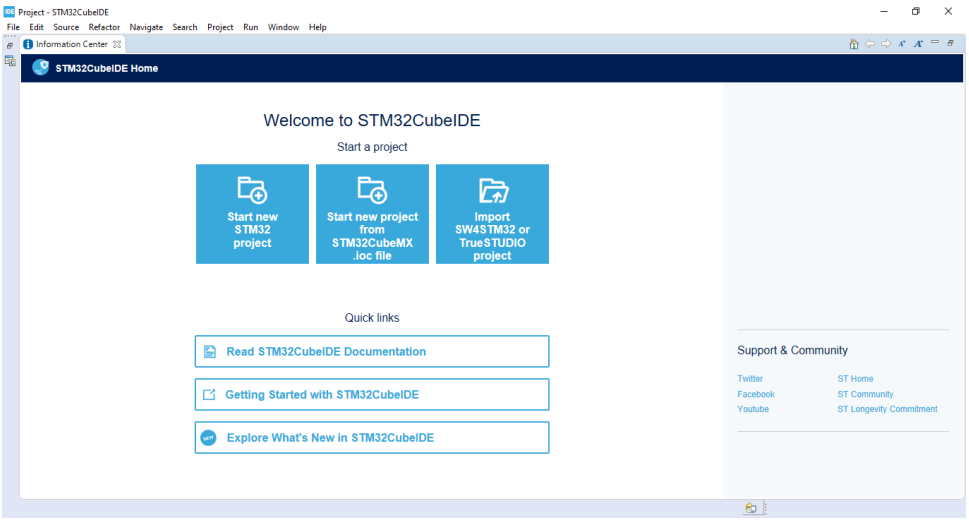


Gambar beserta keterangan diatas merupakan bagian-bagian yang biasanya selalu terpakai jika digunakan untuk membuat simulasi suatu rangkaian. Sedangkan pada gambar dibawah ini banyak digunakan untuk melihat layer atau memindahkan posisi jika dalam posisi zoom dan melihat component list yang dipakai.



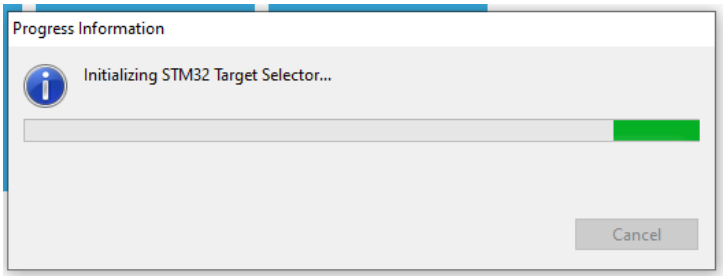
Selain penggunaan proteus untuk praktikum mikrokontroller juga membutuhkan software tambahan untuk memprogram mikrokontroller yaitu menggunakan STM32CUBEIDE dimana merupakan software software yang dirilis langsung oleh STMicroelectronics yang dibangun diatas Eclipse IDE dan tersedia untuk platform windows, mac dan linux. Penggunaan software ini hamper mendukung semua jenis mikrokontroller STM32 kecuali STM32MP1 yang merupakan STM32 dengan MPU pertama.

Gambar Software STM32CubeIDE



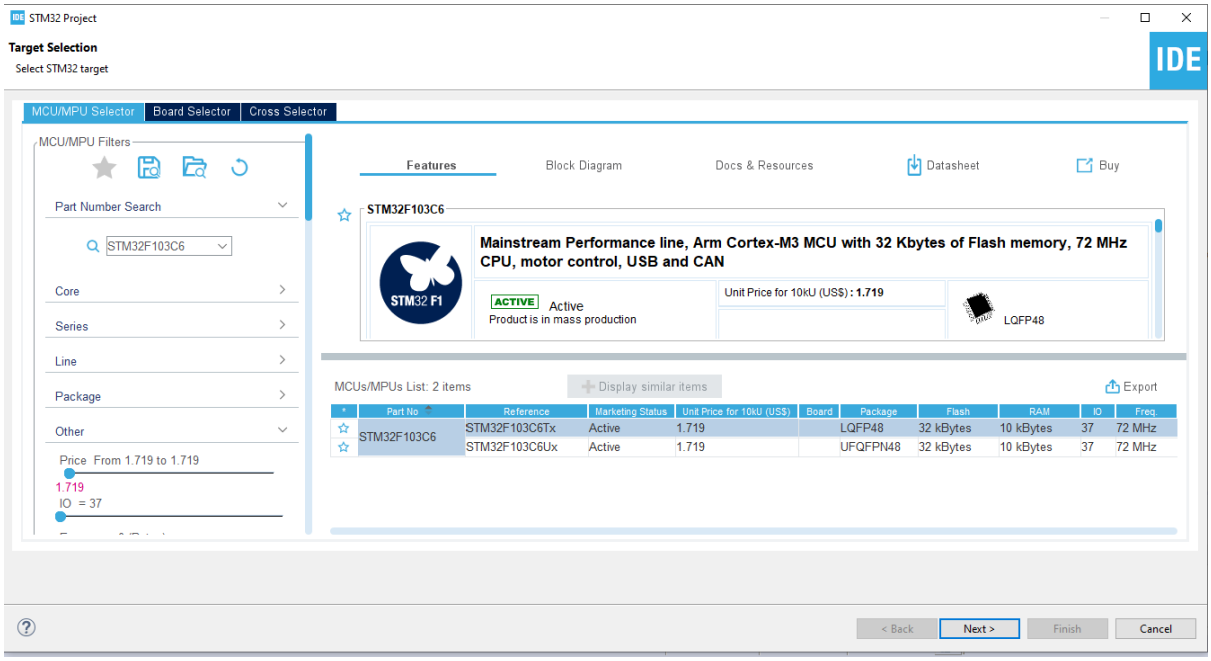
Gambar STM32CubeIDE Home

Pada penggunaan software hampir sama seperti penggunaan software mikrokontroller pada umumnya dimana langkah awal dengan membuat project baru, menentukan perangkat yang ingin dipakai, konfigurasi dan melakukan pemrograman. Penggunaan software ini pertama-tama membuat project baru dimana terdapat pada menu “File” atau pilih icon “Start new STM32 Project” dan setelah itu akan muncul kotak dialog seperti dibawah ini :



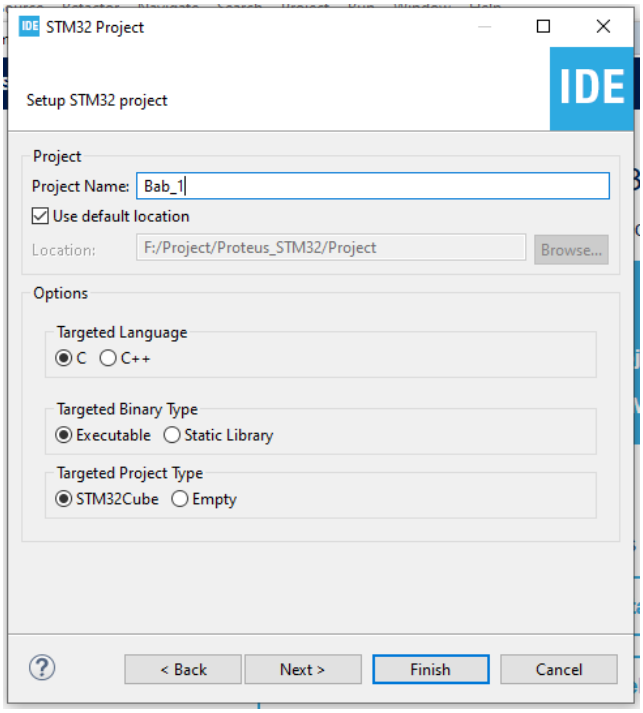
Gambar STM32 Target Selector

Pada gambar diatas untuk membuat project awal sehingga akan muncul target selector untuk inisialisasi dari perangkat STM32 yang akan digunakan. Selanjutnya menunggu saat progress selesai dan akan menampilkan kotak dialog yang digunakan untuk memilih mikrokontroller STM32 yang diinginkan yaitu menggunakan STM32F103C6 seperti gambar dibawah ini dengan mengetik pada kolom seacrh :



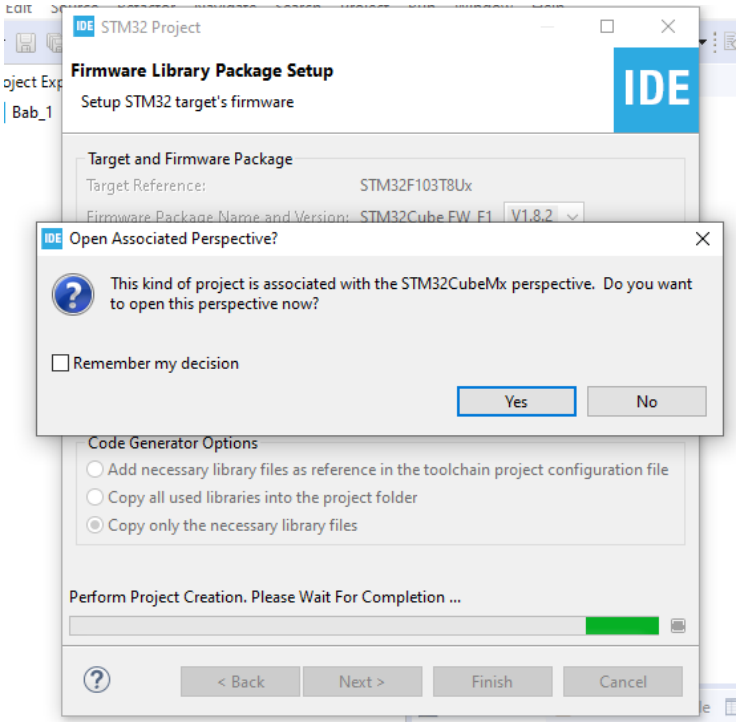
Gambar STM32 Target

Jika telah memilih MCU yang telah sesuai dengan yang dikehendaki maka dapat memilih “Next” dan akan muncul tampilan dimana digunakan untuk memberikan nama dari project yang dinginkan.



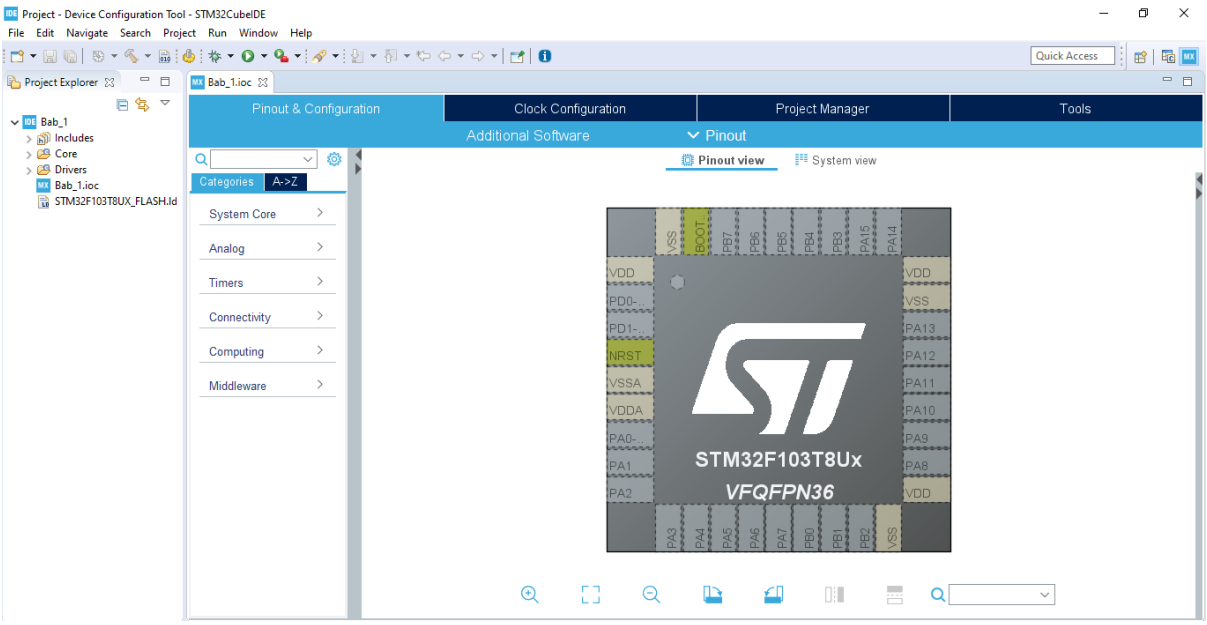
Gambar STM32 Project

Jika telah selesai maka dapat memilih “Finish” dan akan menampilkan persetujuan untuk STM32CubeMx Perspective maka dapat dilanjutkan dengan memilih “Yes” seperti gambar dibawah ini :



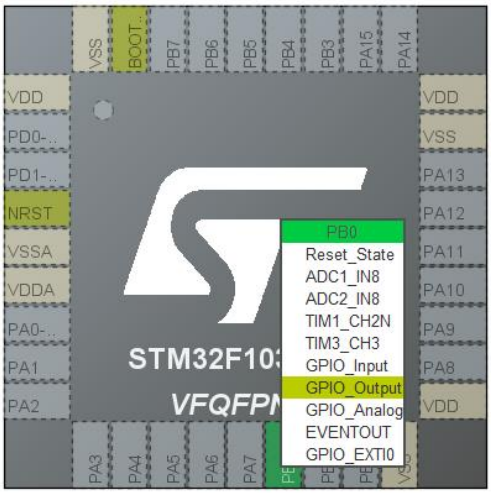
Gambar STM32Cube Perspective

Pada proses selanjutnya akan muncul tampilan project yang telah dibuat dan harus mengkofigurasi dari device yang telah dibuat.



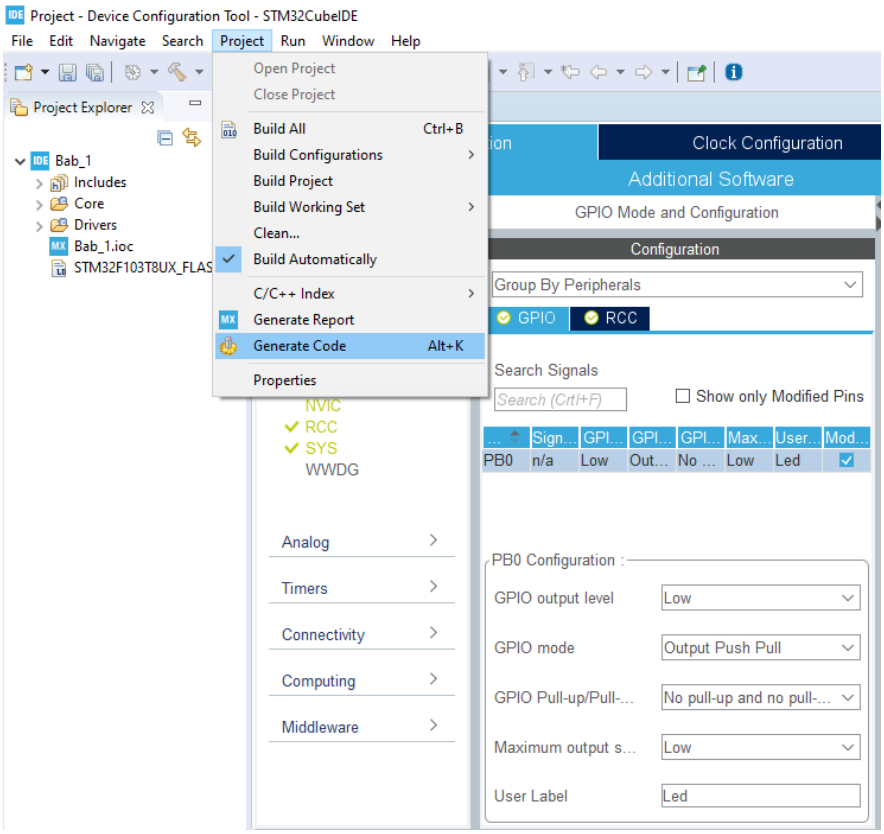
Gambar Project Device Configuration

Pada gambar diatas merupakan tampilan dari device yang kita pilih dimana terdapat “project explorer” untuk mengethaui folder-folder yang ada dalam project yang kita buat yang bertujuan untuk mengetahui library yang digunakan, memprogram maupun menambahkan library yang dibutuhkan. Selain itu juga terdapat tampilan dimana untuk mengatur kegunaan pin dari device yang digunakan dan fasilitas yang disediakan, seperti ADC, SPI, WIRE, dll.



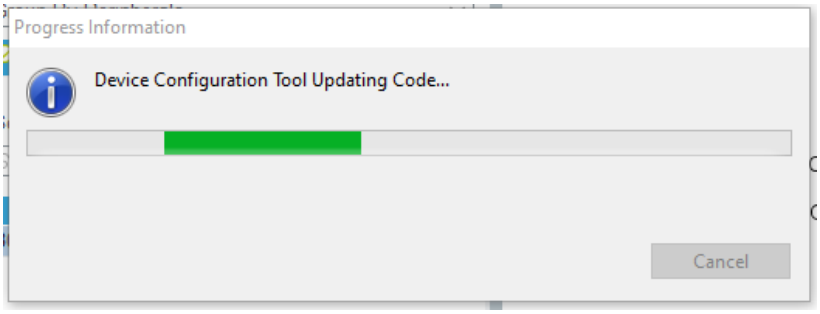
Gambar STM32F103C6

Dalam menentukan kegunaan pin-pin yang tersedia dengan mengecek datasheet sehingga dapat menentukan dengan tepat fungsi kegunaan dari pin yang didesain.



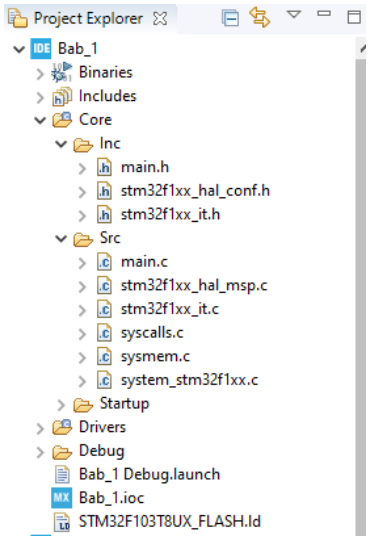
Gambar Generate Code STM32Cube

Jika telah ditentukan dan dikonfigurasi sesuai dengan desain yang dinginkan maka dapat melakukan "Generate Code" untuk melanjutkan ke proses pemrograman device yang telah dikonfigurasi tunggu sampai proses selesai.



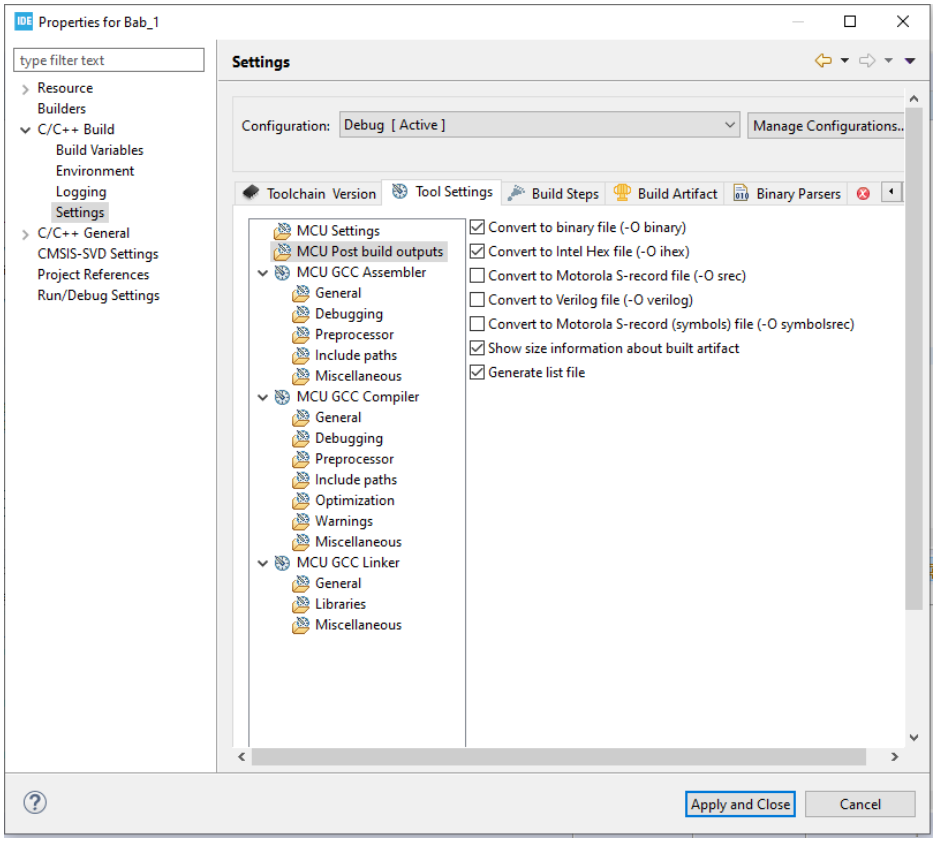
Gambar Tool Updating Code

Jika proses telah selesai maka pada "Project Explorer" dapat dilihat nama project yang telah dibuat dengan beberapa folder-folder pendukung yang memiliki extention .h dan .c yang berisi konfigurasi yang telah dibuat sebelumnya, seperti gambar dibawah ini:



Gambar Folder Project

Pada keterangan sesuai gambar diatas dalam proses menulis program untuk device berada didalam folder “src” dengan file “main.c”. setelah melakukan penulisa program yang dikehendaki langkah selanjutnya merupakan seharunya “Build Project” yang terletak pada menu “Project” tetapi sebelum itu perlu diperhatikan bahwa saat dibuild maka tidak akan memunculkan file .hex untuk keperluan simulasi diproteus sehingga harus diperlukan penambahan pengaturan pada menu “Project – Properties”, maka akan muncul tampilan seperti dibawah ini.



Gambar Pengaturan File Hexa

Setelah muncul tampilan seperti gambar diatas, memilih “C/C++ Build – Setting” dan selanjutnya pilih kembali “Tool – Settings – MCU post build output” dan centang “Convert to intel Hex” setelah selesai pilih “Apply and Close”. Hal ini diperlukan saat project dibuild maka hasil compile akan menghasilkan file .hex yang digunakan untuk simulasi di software proteus.

BAB I

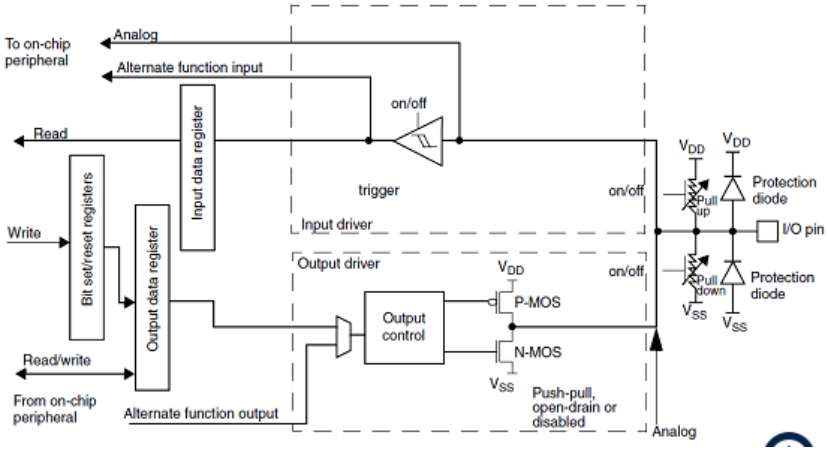
PENGGUNAAN GPIO DAN LED

1.1 Tujuan

Tujuan dari percobaan ini adalah untuk menguji GPIO stm32 sebagai Output untuk menyalakan LED

1.2 Dasar Teori

Pada mikorkontroller ARM penggunaan IO atau yang lebih dikenal dengan GPIO merupakan suatu bagian dari mikrokontroller yang digunakan untuk input maupun output yang berhubungan dengan interface tambahan. Pada masing-masing pin memiliki kegunaan yang berbeda-beda hal ini tergantung dari penggunaan jenis perangkat device STM32 yang digunakan. Maka perlu mengetahui datasheet dari MCU yang digunakan sehingga dapat mengalokasikan pin-pin yang akan digunakan secara tepat.

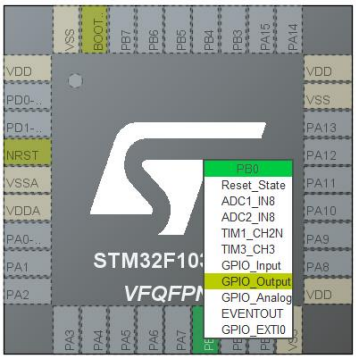


Gambar 1.1 Diagram GPIO STM32

1.3 Langkah Percobaan

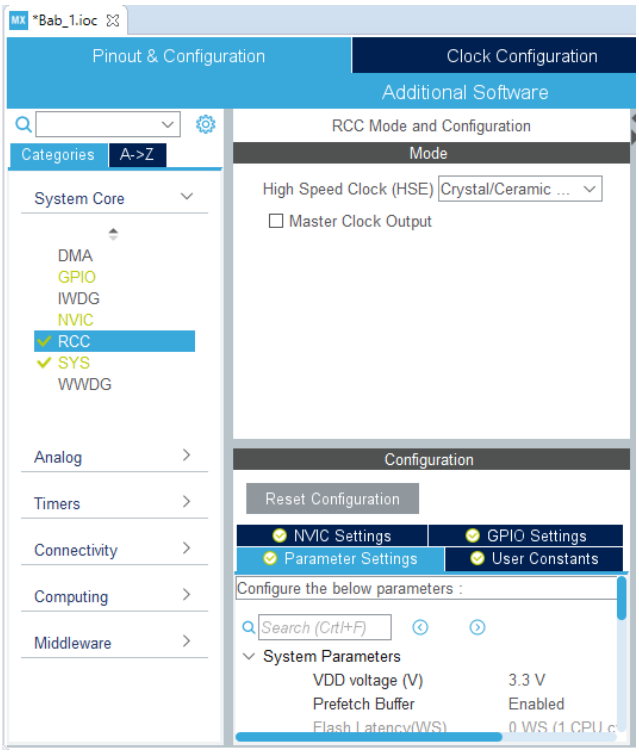
1.3.1. Langkah Percobaan STM32CUBE IDE

- ✓ Membuat project baru menggunakan software STM32CubeIDE
- ✓ Memakai Device STM32F103C6
- ✓ Mengatur PB0 sebagai GPIO_OUTPUT



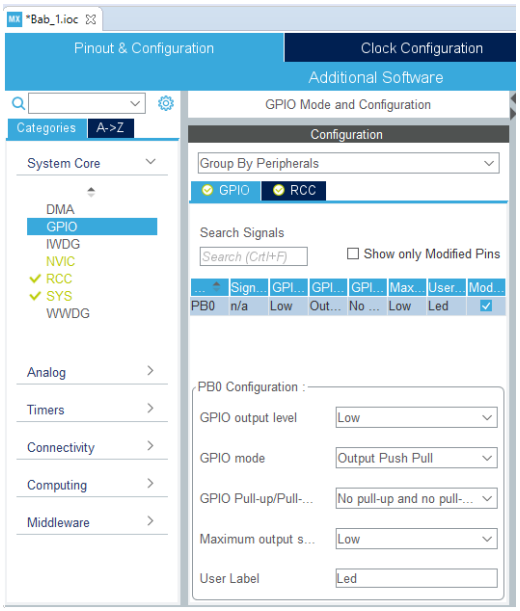
Gambar 1.2 GPIO_OUTPUT

- ✓ Atur RCC – Crystal/Ceramic External



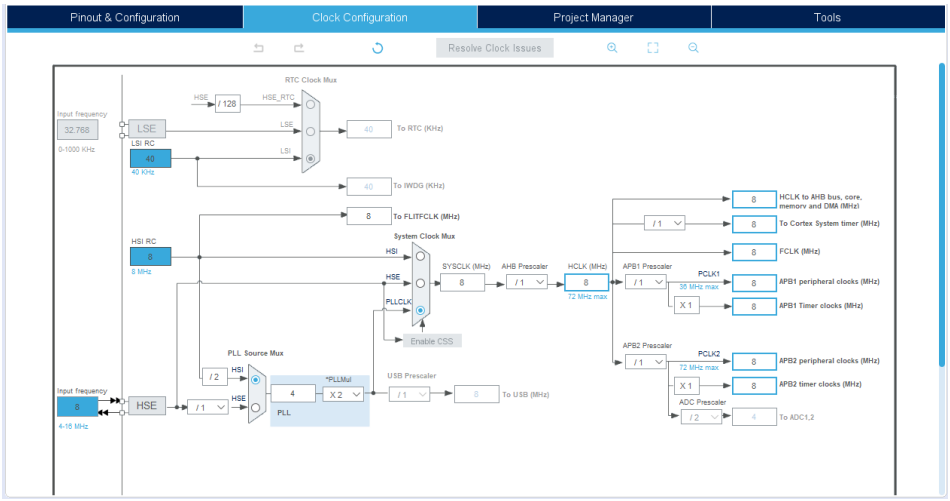
Gambar 1.3 RCC Mode and Configuration

✓ Pilih GPIO dan beri keterangan pada User Label “Led” pada PB0 yang telah dipilih



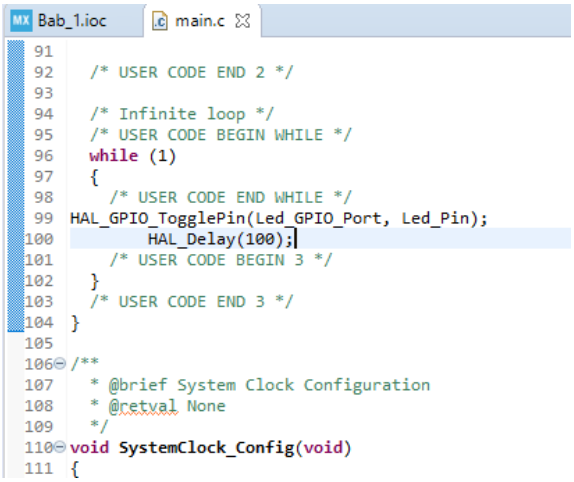
Gambar 1.4 User label GPIO

- ✓ Kemudian pilih menu Clock Configuration dan sesuai kan Item Clock Configuration seperti gambar di bawah ini :



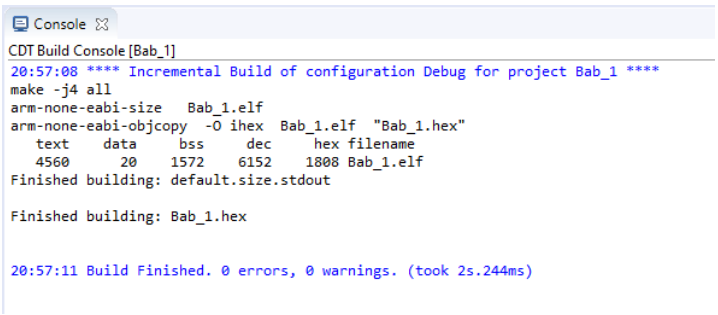
Gambar 1.5 Clock Configuration

- ✓ Setelah selesai pilih generate code sesuai materi dipendahuluan
- ✓ Masuk kedalam Project explorer dan pilih “Core – src – main.c” dan tulis program seperti gambar dibawah ini.



Gambar 1.6 Souce Code Bab 1

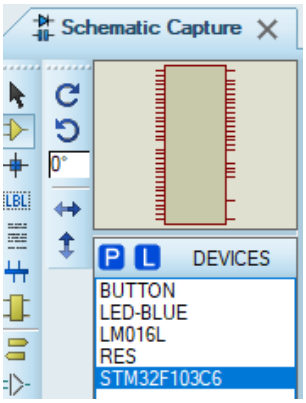
- ✓ Sebelum masuk project build, cek untuk build file .hex sesuai dengan materi pendahuluan
- ✓ Lakukan proses build, pilih menu “Project – Build Project”
- ✓ Perhatikan hasil build (error, warning dan file .hex)



Gambar 1.7 Console Bab 1

1.3.2. Langkah Percobaan Proteus

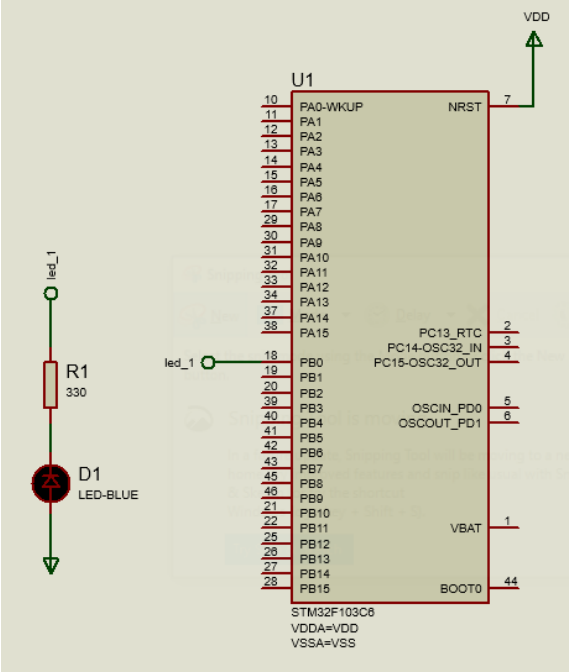
- ✓ Membuat project baru dengan nama “Bab 1” sesuai dengan materi pendahuluan di atas
- ✓ Setelah selesai membuat proct baru dan berada pada tampilan simulasi silanjutnya silahkan memilih Component Mode – Pick from Library
- ✓ Maka akan muncul seperti gambar silahkan isi keywords dengan mengetik “LED-BLUE” lalu “Double Click”.



Gambar 1.7 Component list

- ✓ Setelah itu silahkan ditambahkan sesuai dengan komponen list yang diinginkan.
- ✓ Jika benar maka akan muncul komponen pada component list

✓ Selanjutnya buat lah rangkaian seperti gambar di bawah ini



Gambar 1.8 Rangkaian Bab 1

✓ Setelah selesai merangkai sesuai gambar diatas dapat langsung menjalankan proses simulasi untuk mengetahui hasilnya dengan cara pilih tombol “Run the Simulation”  terletak kiri bawah.

✓ Untuk mengganti nilai dari komponen dapat dengan meng-klik 2 pada gambar komponen tersebut.

1.4 Analisa Percobaan

1.5 Kesimpulan

BAB II

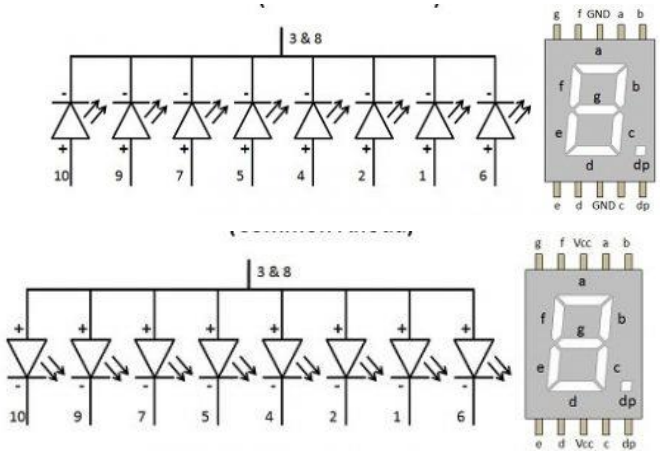
7-SEGMENT

2.1 Tujuan

Mengetahui penggunaan perangkat 7-segment pada mikrokontroller STM32

2.2 Dasar Teori

Seven segment merupakan bagian-bagian yang digunakan untuk menampilkan angka atau bilangan decimal. Seven segment tersebut terbagi menjadi 7 batang LED yang disusun membentuk angka 8 dengan menggunakan huruf a-f yang disebut DOT MATRIKS. Setiap segment ini terdiri dari 1 atau 2 LED (Light Emitting Dioda). Seven segment bisa menunjukan angka-angka desimal serta beberapa bentuk tertentu melalui gabungan aktif atau tidaknya LED penyususnan dalam seven segment



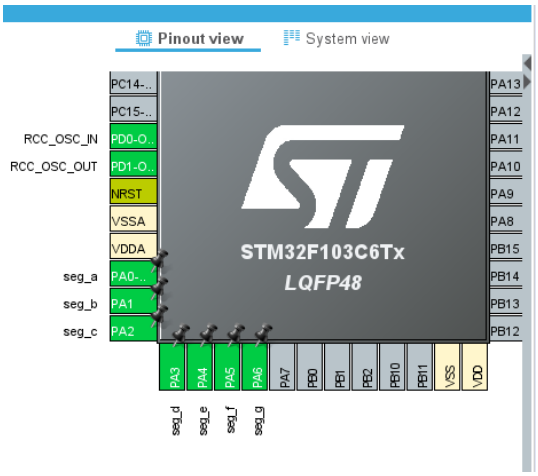
Gambar 2.1 Seven Segment Common anode dan cathode

Seven Segment Display sering dipakai oleh para peminat Elektronika adalah 7 bagian yang memakai LED (Light Emitting Diode) sebagai penerangnya. LED 7 Segmen tersebut biasanya mempunyai 7 Segmen atau elemen garis dan 1 segmen titik yang menandakan “koma” Desimal. Jumlah semua segmen atau elemen LED sebenarnya adalah 8

2.3 Langkah Percobaan

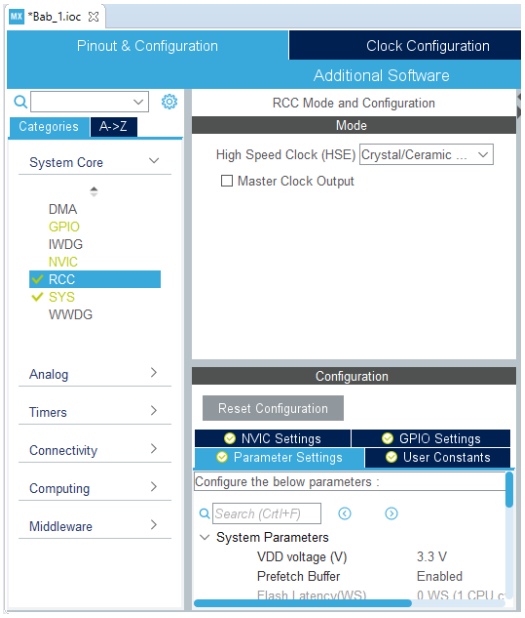
2.3.1. Langkah Percobaan STM32CUBE IDE

- ✓ Membuat project baru menggunakan software STM32CubeIDE
- ✓ Memakai Device STM32F103C6
- ✓ Mengatur PA0-PA6 sebagai GPIO_OUTPUT



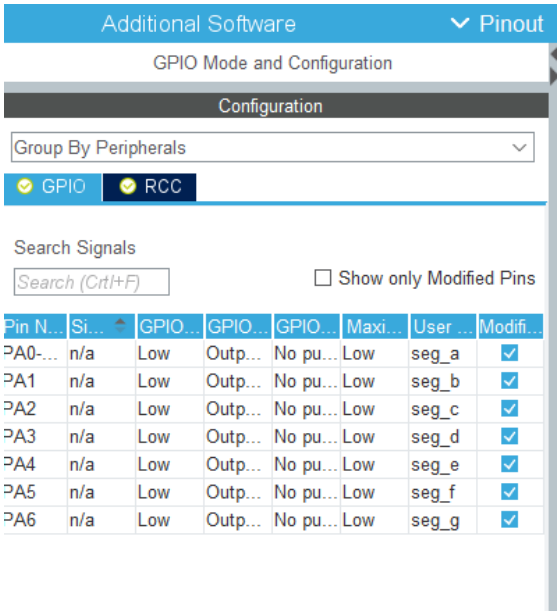
Gambar 2.2 GPIO_OUTPUT

✓ Atur RCC – Crystal/Ceramic External



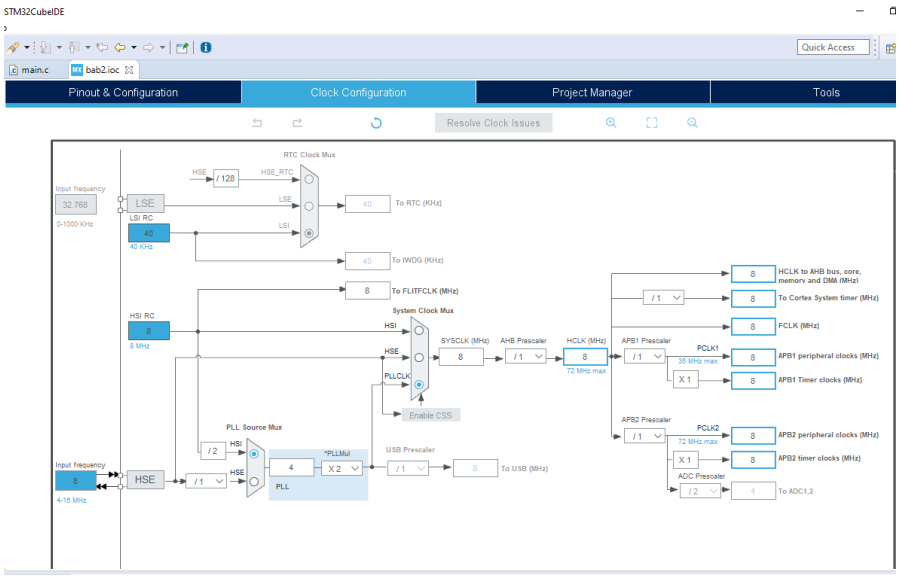
Gambar 2.3 RCC Mode and Configuration

✓ Pilih GPIO dan beri keterangan pada User Label sesuai gambar dibawah ini.



Gambar 2.4 User Label GPIO

✓ Kemudian pilih menu Clock Configuration dan sesuai kan Item Clock Configuration seperti gambar di bawah ini :



Gambar 2.5 Clock Configuration

✓ Setelah selesai pilih generate code sesuai materi dipendahuluan

✓ Masuk kedalam Project explorer dan pilih “Core – src – main.c” dan tulis program seperti gambar dibawah ini.

Bab_3_segment.ioc | main.c

```
/* Private function prototypes -----------------
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
/* USER CODE BEGIN PFP */

/* USER CODE END PFP */

/* Private user code ---------------------------
/* USER CODE BEGIN 0 */
uint16_t T[16] ={
        0x00C0,  //0
        0x00F9,  //1
        0x00A4,  //2
        0x00B0,  //3
        0x0099,  //4
        0x0092,  //5
        0x0082,  //6
        0x00F8,  //7
        0x0080,  //8
        0x0090,  //9
        0x0088,  //A
        0x0083,  //B
        0x00C6,  //C
        0x00A1,  //D
        0x0086,  //E
        0x008E  //F
};
/* USER CODE END 0 */

/**
  * @brief  The application entry point.
  * @retval int
  */
```

Bab_3_segment.ioc | main.c

```
  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  /* USER CODE BEGIN 2 */

  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */
      int i;
      for(i=0;i<16;i++){
          GPIOA->ODR =T[i];
          HAL_Delay(100);
      }
    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}

/**
  * @brief System Clock Configuration
  * @retval None
  */
void SystemClock_Config(void)
```

Gambar 2.6 Sorce Code Bab 2

- ✓ Lakukan proses build, pilih menu “Project – Build Project”
- ✓ Perhatikan hasil build (error, warning dan file .hex)

Console

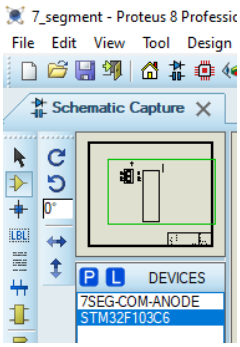
CDT Build Console [Bab_3_segment]

```
arm-none-eabi-size   Bab_3_segment.elf
   text    data     bss     dec     hex filename
   4528      52    1572    6152    1808 Bab_3_segment.elf
Finished building: default.size.stdout
Finished building: Bab_3_segment.bin
Finished building: Bab_3_segment.hex


Finished building: Bab_3_segment.list


09:34:02 Build Finished. 0 errors, 0 warnings. (took 5s.592ms)
```

Gambar 2.7 Console Bab 2

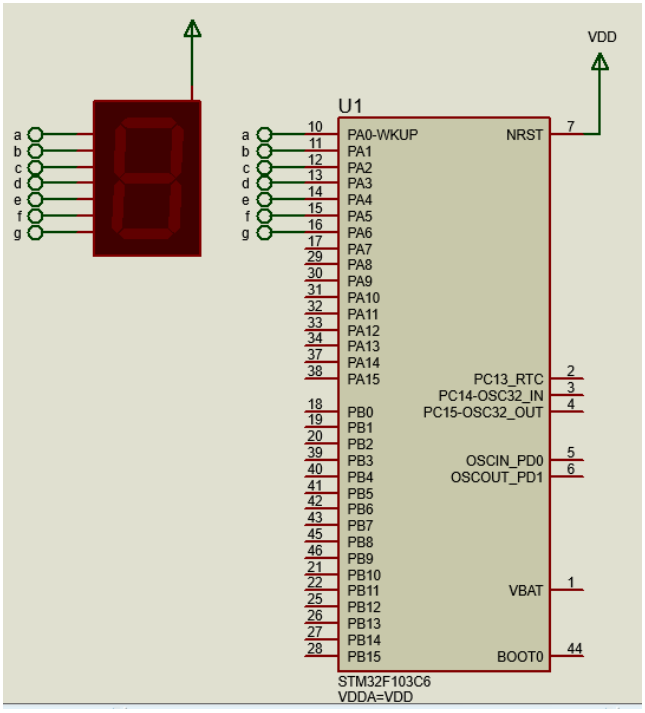
2.3.2. Langkah Percobaan Proteus

- ✓ Membuat project baru dengan nama “Bab 2” sesuai dengan materi pendahuluan di atas
- ✓ Setelah selesai membuat proct baru dan berada pada tampilan simulasi silanjutnya silahkan memilih Component Mode – Pick From Library
- ✓ Maka akan muncul seperti gambar silahkan isi keywords dengan mengetik “7SEG-COM-ANODE” lalu “Double Click”.



Gambar 2.7 Component list

- ✓ Setelah itu silahkan ditambahkan sesuai dengan komponen list yang diinginkan.
- ✓ Jika benar maka akan muncul komponen pada component list
- ✓ Selanjutnya buat lah rangkaian seperti gambar di bawah ini



Gambar 2.8 Rangkaian Bab 2

- ✓ Untuk penambahan alat ukur voltmeter maupun amperemeter berada dalam “virtual instrument mode” pilih DC Voltmeter

- ✓ Setelah selesai merangkai sesuai gambar diatas dapat langsung menjalankan proses simulasi untuk mengetahui hasilnya dengan cara pilih tombol “Run the Simulation”  terletak kiri bawah.
- ✓ Untuk mengganti nilai dari komponen dapat dengan meng-klik 2 pada gambar komponen tersebut.

2.4 Analisa Percobaan
2.5 Kesimpulan

BAB III

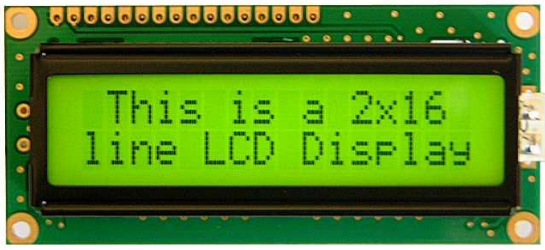
LCD

3.1 Tujuan

Mengetahui cara menggunakan LCD 16x2 pada mikrokontroller STM32F1

3.2 Dasar Teori

Teknologi Display LCD ini memungkinkan produk-produk elektronik dibuat menjadi jauh lebih tipis jika dibanding dengan teknologi Tabung Sinar Katoda (*Cathode Ray Tube* atau CRT). Jika dibandingkan dengan teknologi CRT, LCD juga jauh lebih hemat dalam mengkonsumsi daya karena LCD bekerja berdasarkan prinsip pemblokiran cahaya sedangkan CRT berdasarkan prinsip pemancaran cahaya. Namun LCD membutuhkan lampu backlight (cahaya latar belakang) sebagai cahaya pendukung karena LCD sendiri tidak memancarkan cahaya. Beberapa jenis backlight yang umum digunakan untuk LCD diantaranya adalah backlight CCFL (*Cold cathode fluorescent lamps*) dan backlight LED (*Light-emitting diodes*).



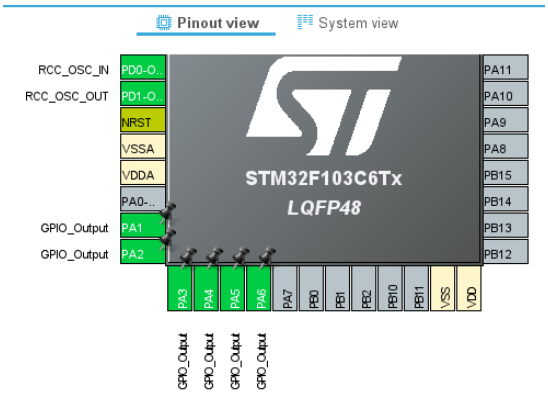
Gambar 3.1 LCD 16x2

LCD (Liquid Cristal Display) berfungsi untuk menampilkan karakter angka, huruf ataupun simbol dengan lebih baik dan dengan konsumsi arus yang rendah. LCD (Liquid Cristal Display) dot matrik M1632 merupakan modul LCD buatan hitachi. Modul LCD (Liquid Cristal Display) dot matrik M1632 terdiri dari bagian penampil karakter (LCD) yang berfungsi menampilkan karakter dan bagian sistem prosesor LCD dalam bentuk modul dengan mikrokontroler yang diletakan dibagian belakan LCD tersebut yang berfungsi untuk mengatur tampilan LCD serta mengatur komunikasi antara LCD dengan mikrokontroler yang menggunakan modul LCD tersebut. LCD M1632 merupakan modul LCD dengan tampilan 2×16 (2 baris x 16 kolom) dengan konsumsi daya rendah

3.3 Langkah Percobaan

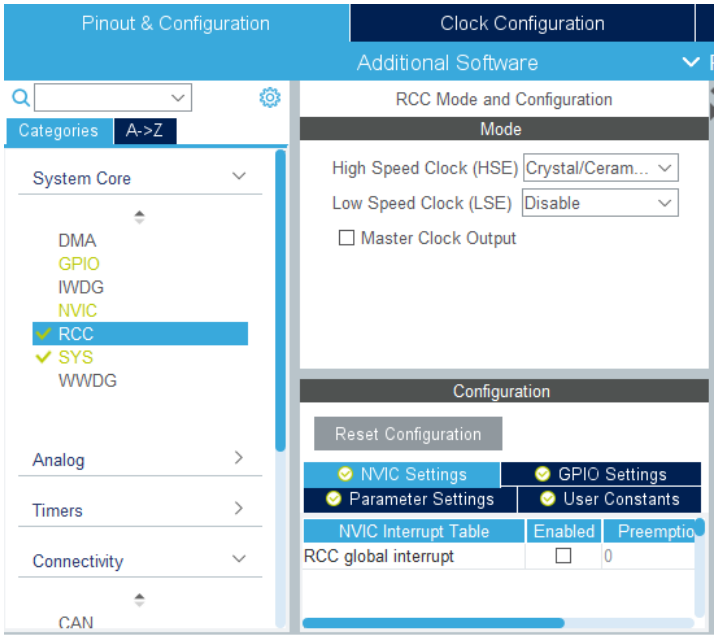
3.3.1. Langkah Percobaan STM32CUBE IDE

- ✓ Membuat project baru menggunakan software STM32CubeIDE
- ✓ Memakai Device STM32F103C6
- ✓ Mengatur PortA sesuai dengan gambar dibawah ini



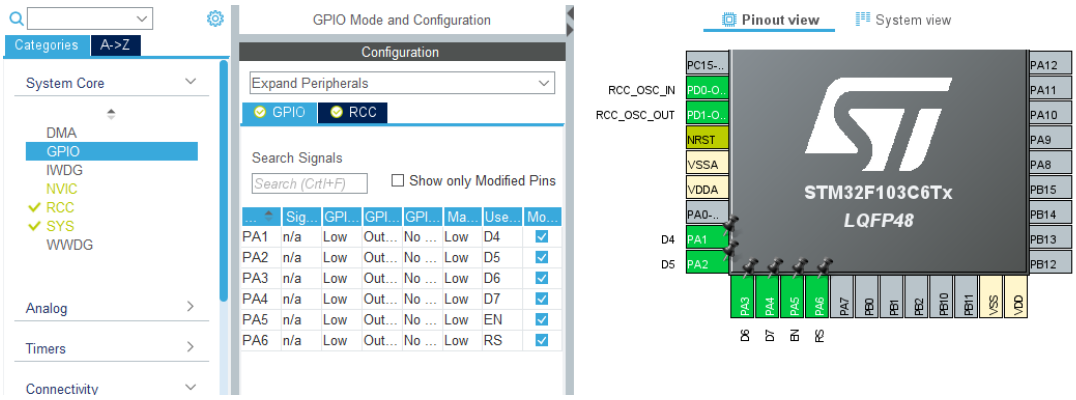
Gambar 3.2 GPIO_OUTPUT

✓ Atur RCC – Crystal/Ceramic



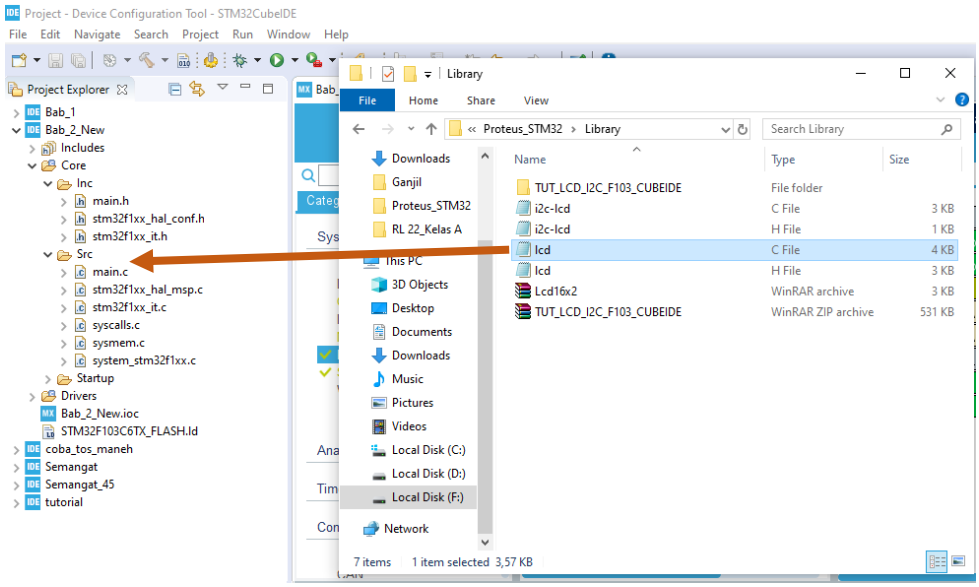
Gambar 3.3 RCC Mode And Configuration

✓ Pilih GPIO dan beri keterangan pada User Label “D4” pada PA1 yang telah dipilih, dan disesuaikan dengan gambar dibawah ini.



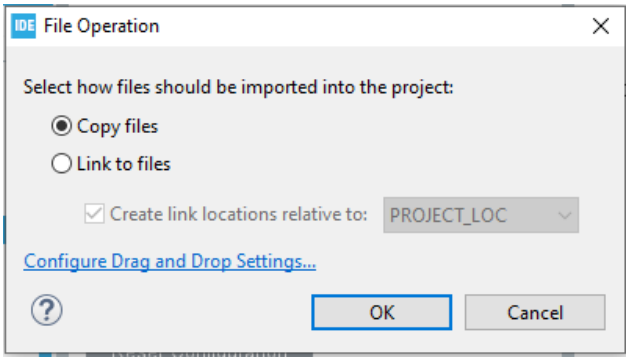
Gambar 3.4 User Label GPIO

- ✓ Setelah selesai pilih generate code sesuai materi dipendahuluan
- ✓ Selanjutnya masukkan library tambah “LCD” dengan cara drag and drop file ke dalam folder yang dibutuhkan sesuai dengan extention file.



Gambar 3.5 Library LCD

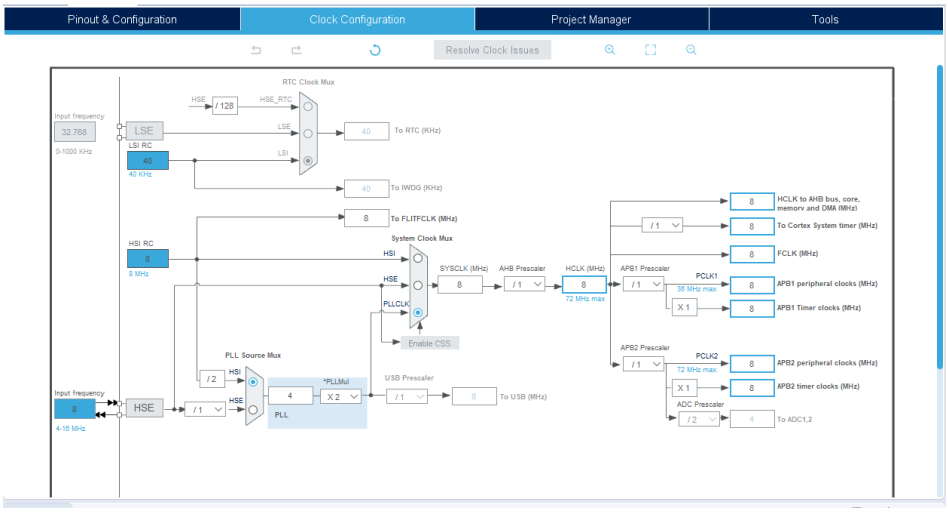
- ✓ Pilih “OK”



Gambar 3.6 File Operation

- ✓ Ulangi Langkah tersebut untuk memasukkan library LCD dengan extention .h dengan cara yang sama seperti sebelumnya.

✓ Kemudian pilih menu Clock Configuration dan sesuai kan Item Clock Configuration seperti gamabr di bawah ini :



Gambar 3.7 Clock Configuration

✓ Masuk kedalam Project explorer dan pilih “Core – src – main.c” dan tulis program seperti gambar dibawah ini.

main.c | lcd.c | Bab_3_LCD.ioc

```
88
89    /* Initialize all configured peripherals */
90    MX_GPIO_Init();
91    /* USER CODE BEGIN 2 */
92    Lcd_PortType ports[] = {
93                D4_GPIO_Port, D5_GPIO_Port, D6_GPIO_Port, D7_GPIO_Port
94              };
95
96              Lcd_PinType pins[] = {
97                  D4_Pin, D5_Pin, D6_Pin, D7_Pin
98              };
99
100             Lcd_HandleTypeDef lcd = Lcd_create(ports, pins, RS_GPIO_Port, RS_Pin, EN_GPIO_Port, EN_Pin, LCD_4_BIT_MODE);
101   /* USER CODE END 2 */
102
103   /* Infinite loop */
104   /* USER CODE BEGIN WHILE */
105   while (1)
106   {
107     /* USER CODE END WHILE */
108     Lcd_cursor(&lcd, 0,0);
109               Lcd_string(&lcd, "Robotikid");
110               Lcd_clear(&lcd);
111
112
113               Lcd_cursor(&lcd, 0,0);
114               Lcd_int(&lcd, 100);
115               HAL_Delay(50);
116   /* USER CODE BEGIN 3 */
117   }
118   /* USER CODE END 3 */
119 }
```

Gambar 3.8 Source Code Bab 3

✓ Lakukan proses build, pilih menu “Project – Build Project”

✓ Perhatikan hasil build (error, warning dan file .hex)

Console

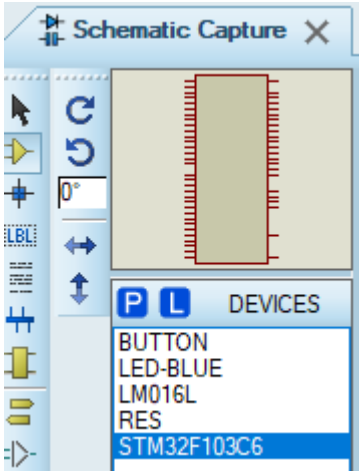
```
CDT Build Console [Bab_3_LCD]
   7576     120    1584    9280    2440 Bab_3_LCD.elf
Finished building: default.size.stdout
Finished building: Bab_3_LCD.bin

Finished building: Bab_3_LCD.list
Finished building: Bab_3_LCD.hex


11:09:12 Build Finished. 0 errors, 0 warnings. (took 6s.681ms)
```

Gambar 3.9 Console Bab 3

- ✓ Abaikan jika terdapat warning.

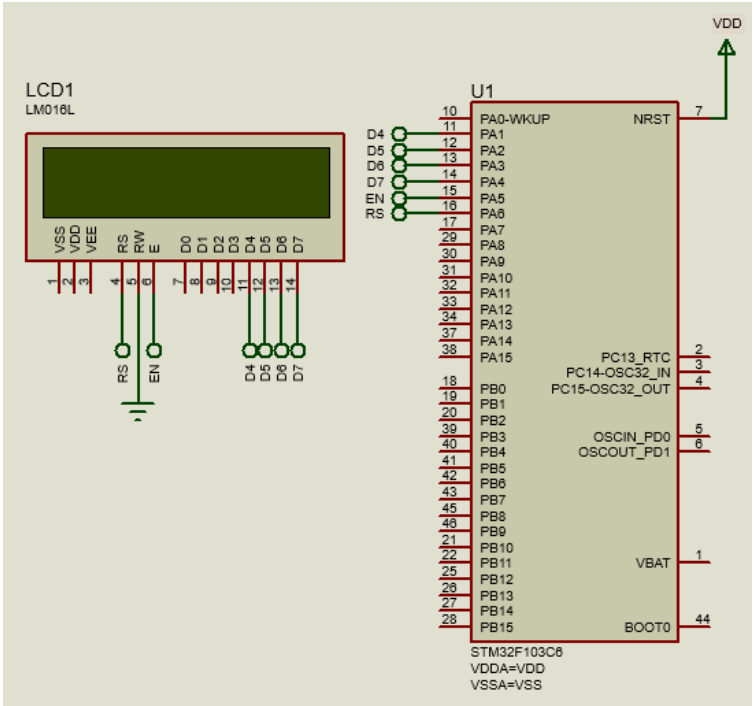
3.3.2. Langkah Percobaan Proteus

- ✓ Membuat project baru dengan nama “Bab 3” sesuai dengan materi pendahuluan di atas
- ✓ Setelah selesai membuat proct baru dan berada pada tampilan simulasi silanjutnya silahkan memilih Component Mode – Pick From Library
- ✓ Maka akan muncul seperti gambar silahkan isi keywords dengan mengetik “LM016L” lalu “Double Click”.



Gambar 3.10 Component list

- ✓ Setelah itu silahkan ditambahkan sesuai dengan komponen list yang diinginkan.
- ✓ Jika benar maka akan muncul komponen pada component list
- ✓ Selanjutnya buat lah rangkaian seperti gambar di bawah ini



Gambar 3.11 Rangkain Bab 3

- ✓ Setelah selesai merangkai sesuai gambar diatas dapat langsung menjalankan proses simulasi untuk mengetahui hasilnya dengan cara pilih tombol “Run the Simulation”  terletak kiri bawah.
- ✓ Untuk mengganti nilai dari komponen dapat dengan meng-klik 2 pada gambar komponen tersebut.

3.4 Analisa Percobaan
3.5 Kesimpulan

BAB IV

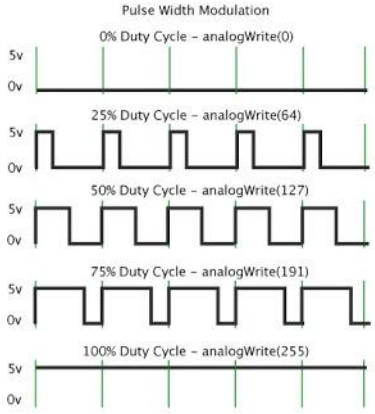
PWM DENGAN INTERNAL TIMER

4.1 Tujuan

Tujuan dari percobaan ini adalah untuk menguji pembangkit PWM dengan internal timer pada STM32

4.2 Dasar Teori

PWM adalah kepanjangan dari *Pulse Width Modulation* atau dalam bahasa Indonesia dapat diterjemahkan menjadi Modulasi Lebar Pulsa. Jadi pada dasarnya, PWM adalah suatu teknik modulasi yang mengubah lebar pulsa (pulse width) dengan nilai frekuensi dan amplitudo yang tetap. PWM dapat dianggap sebagai kebalikan dari ADC (Analog to Digital Converter) yang mengkonversi sinyal Analog ke Digital, PWM atau Pulse Width Modulation ini digunakan menghasilkan sinyal analog dari perangkat Digital (contohnya dari Mikrokontroller)



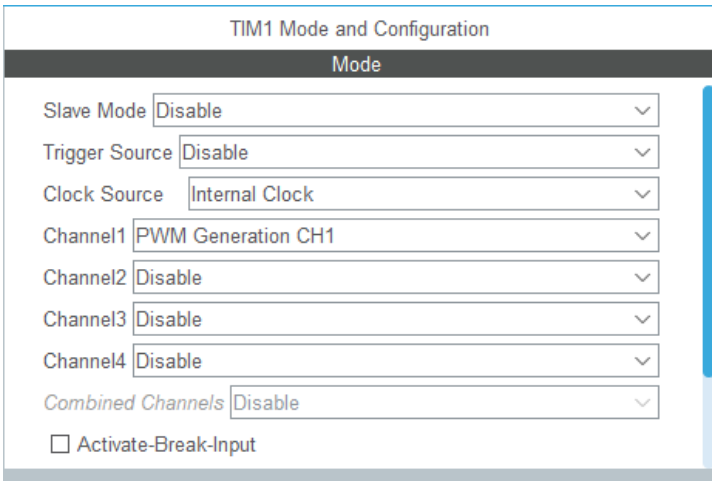
Gambar 4.1 Duty Cycle

Persentase waktu di mana sinyal PWM tetap pada kondisi TINGGI (ON Time) disebut dengan "siklus kerja" atau "*Duty Cycle*". Kondisi yang sinyalnya selalu dalam kondisi ON disebut sebagai 100% *Duty Cycle* (Siklus Kerja 100%), sedangkan kondisi yang sinyalnya selalu dalam kondisi OFF (mati) disebut dengan 0% *Duty Cycle* (Siklus Kerja 0%).

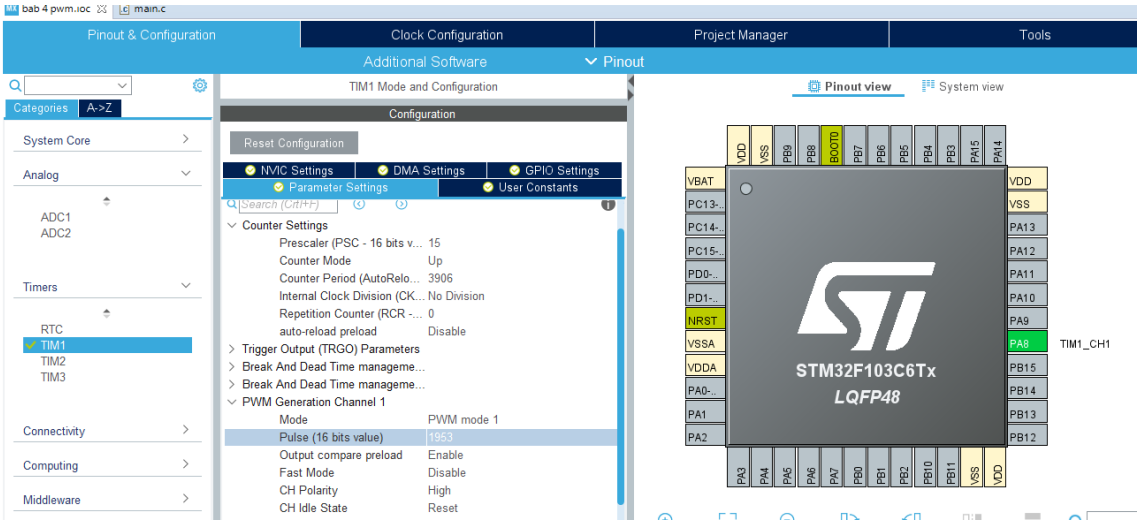
4.3 Langkah Percobaan

4.3.1. Langkah Percobaan STM32CUBE IDE

- ✓ Membuat project baru menggunakan software STM32CubeIDE
- ✓ Memakai Device STM32F103C6
- ✓ Pilih kebutuhan internal TIMER dengan memilih TIM1 dan mengatur Mode serta Parameters Settings seperti gambar dibawah ini.

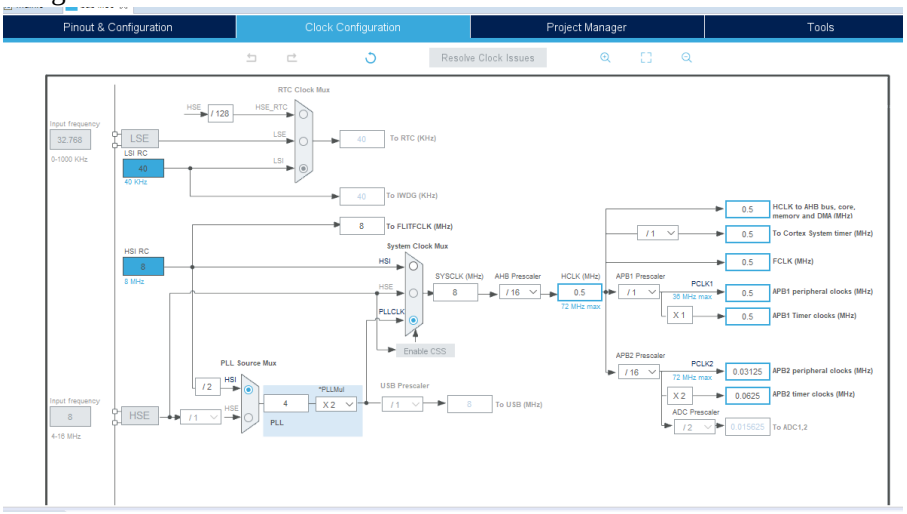


Gambar 4.2 TIM1 Mode and Configuration



Gambar 4.3 Parameter Setting TIM1

✓ Kemudian pilih menu Clock Configuration dan sesuai kan Item Clock Configuration seperti gamabr di bawah ini :



Gambar 4.4 Clock Configuration

✓ Setelah selesai pilih generate code sesuai materi dipendahuluan

- ✓ Masuk kedalam Project explorer dan pilih “Core – src – main.c” dan tulis program seperti gambar dibawah ini.

```
  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_TIM1_Init();
  /* USER CODE BEGIN 2 */
  HAL_TIM_Base_Start(&htim1);
  HAL_TIM_PWM_Start(&htim1, TIM_CHANNEL_1);
  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}
```

Gambar 4.5 Source Code Bab 4

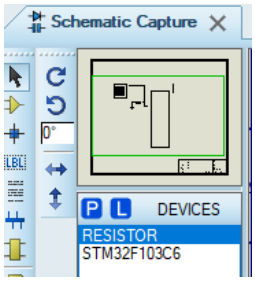
- ✓ Lakukan proses build, pilih menu “Project – Build Project”
- ✓ Perhatikan hasil build (error, warning dan file .hex)

```
Console
CDT Build Console [bab 4 pwm]
arm-none-eabi-objcopy  -O binary  bab\ 4\ pwm.elf  "bab 4 pwm.bin"
arm-none-eabi-size   bab\ 4\ pwm.elf
   text    data     bss     dec     hex filename
   7992      20    1644    9656    25b8 bab 4 pwm.elf
Finished building: default.size.stdout

Finished building: bab 4 pwm.hex
Finished building: bab 4 pwm.bin

Finished building: bab 4 pwm.list


10:15:01 Build Finished. 0 errors, 0 warnings. (took 4s.243ms)
```

Gambar 4.6 Console Bab 4

4.3.2. Langkah Percobaan Proteus

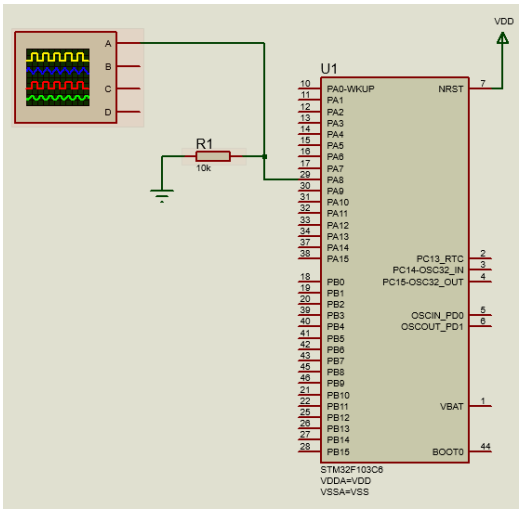
- ✓ Membuat project baru dengan nama “Bab 4” sesuai dengan materi pendahuluan di atas
- ✓ Setelah selesai membuat proct baru dan berada pada tampilan simulasi silanjutnya silahkan memilih Component Mode – Pick From Library
- ✓ Maka akan muncul seperti gambar silahkan isi keywords dengan mengetik “RESISTOR” lalu “Double Click”.



Gambar 4.7 Component list

- ✓ Setelah itu silahkan ditambahkan sesuai dengan komponen list yang diinginkan.
- ✓ Jika benar maka akan muncul komponen pada component list

- ✓ Selanjutnya buat lah rangkaian seperti gambar di bawah ini



Gambar 4.8 rangkain Bab 4

- ✓ Untuk penambahan alat ukur OSCILOSCOPE berada dalam “virtual instrument mode”
- ✓ Setelah selesai merangkai sesuai gambar diatas dapat langsung menjalankan proses simulasi untuk mengetahui hasilnya dengan cara pilih tombol “Run the Simulation”  terletak kiri bawah.
- ✓ Untuk mengganti nilai dari komponen dapat dengan meng-klik 2 pada gambar komponen tersebut.

4.4 Analisa Percobaan
4.5 Kesimpulan

BAB V

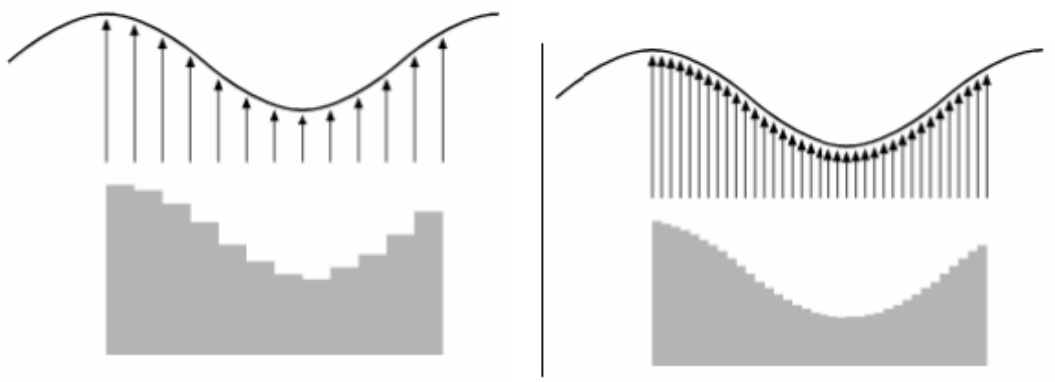
ANALOD TO DIGITAL CONVERTER

5.1 Tujuan

Tujuan dari percobaan ini adalah untuk menguji nilai Bit ADC pada STM32

5.2 Dasar Teori

Analog To Digital Converter (ADC) adalah pengubah input analog menjadi kode – kode digital. ADC banyak digunakan sebagai Pengatur proses industri, komunikasi digital dan rangkaian pengukuran/ pengujian. Umumnya ADC digunakan sebagai perantara antara sensor yang kebanyakan analog dengan sistim komputer seperti sensor suhu, cahaya, tekanan/ berat, aliran dan sebagainya kemudian diukur dengan menggunakan sistim digital (komputer). ADC (Analog to Digital Converter) memiliki 2 karakter prinsip, yaitu kecepatan sampling dan resolusi. Kecepatan sampling suatu ADC menyatakan seberapa sering sinyal analog dikonversikan ke bentuk sinyal digital pada selang waktu tertentu. Kecepatan sampling biasanya dinyatakan dalam sample per second (SPS).



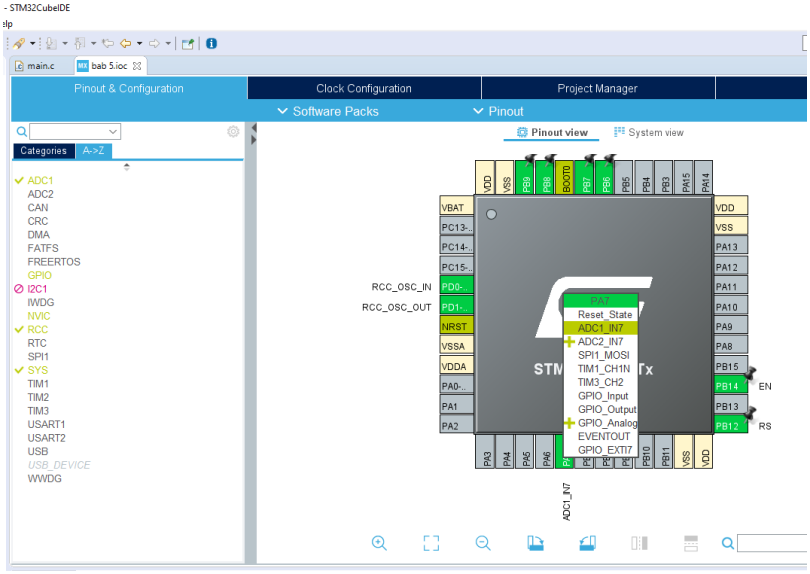
Gambar 5.1 ADC Dengan kecepatan Sampling Rendah dan kecepatan Sampling tinggi

Resolusi ADC menentukan ketelitian nilai hasil konversi ADC. Sebagai contoh: ADC 8 bit akan memiliki output 8 bit data digital, ini berarti sinyal input dapat dinyatakan dalam 255 (2n – 1) nilai diskrit. ADC 12 bit memiliki 12 bit output data digital, ini berarti sinyal input dapat dinyatakan dalam 4096 nilai diskrit. Dari contoh diatas ADC 12 bit akan memberikan ketelitian nilai hasil konversi yang jauh lebih baik daripada ADC 8 bit.

5.3 Langkah Percobaan

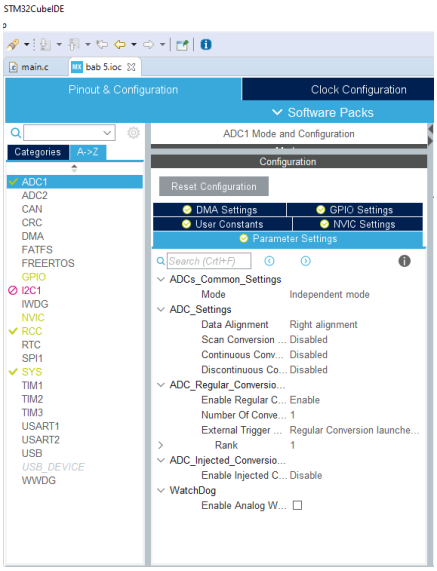
5.3.1. Langkah Percobaan STM32CUBE IDE

- ✓ Membuat project baru menggunakan software STM32CubeIDE
- ✓ Memakai Device STM32F103C6
- ✓ Mengatur PA7 sebagai ADC1_IN7



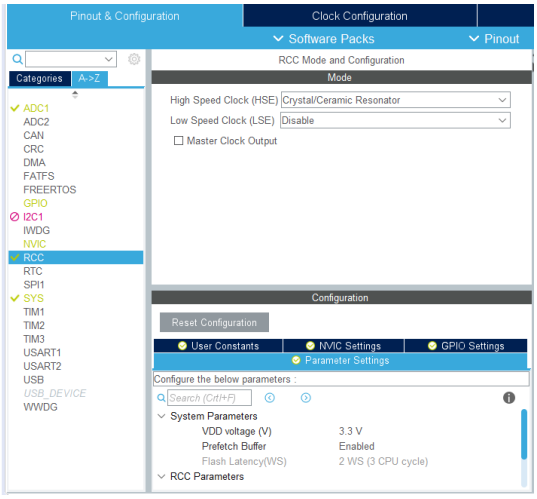
Gambar 5.2 ADC1 Mode and Configuration

✓ Sesuai kan Item pada Configuration pada ADC1_IN7 seperrti pada gambar di bawah ini:



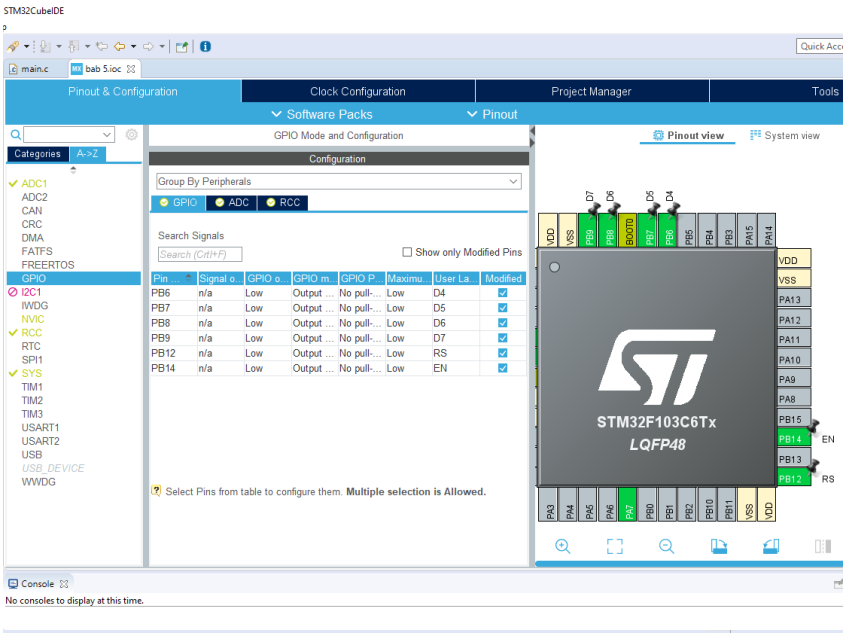
Gambar 5.3 Parameter Setting ADC1

✓ Pilih menu RCC,dan ubah pada HSE dari Disable ke rystal/Ceramic Res, seperrti pada gambar di bawah ini:



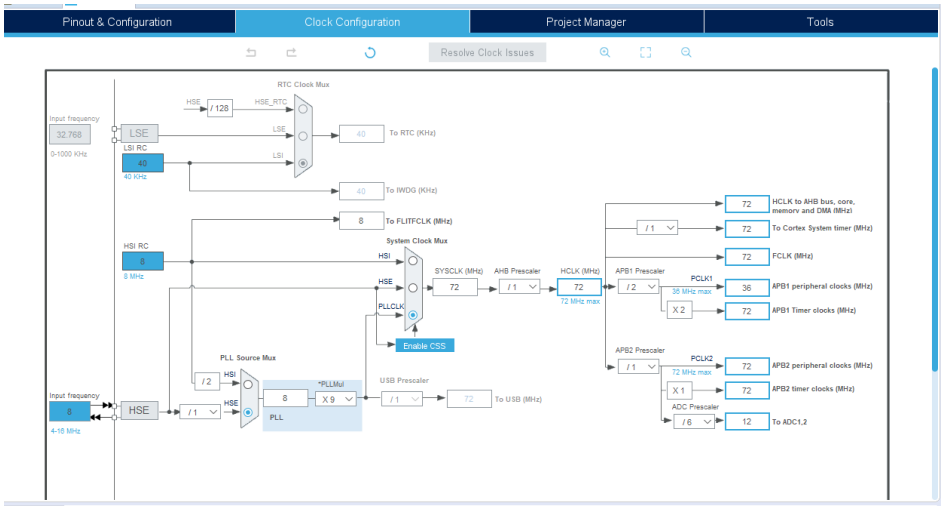
Gambar 5.4 RCC Mode and Configuration

✓ Atur PB6 PB7, PB8, PB9 PB12 PB14 Menjadi GPIO_OUTPUT dan beri label pada pin tersebut sesuai gambar di bawah ini :



Gambar 5.5 User Label GPIO

✓ Kemudian pilih menu Clock Configuration dan sesuai kan Item Clock Configuration seperti gamabr di bawah ini :



Gambar 5.6 Clock Configuration

✓ Setelah selesai pilih generate code sesuai materi dipendahuluan

✓ Masuk kedalam Project explorer dan pilih “Core – src – main.c” dan tulis program seperti gambar dibawah ini dan jangan lupa masukan library LCD seperti bab 3:

```
#include "main.h"
#include "lcd.h"
#include "stdio.h"

ADC_HandleTypeDef hadc1; //
Lcd_HandleTypeDef lcd;

int main(void)
{

uint16_t adc_value = 0;

  HAL_Init();


  SystemClock_Config();
  MX_GPIO_Init();
  Lcd_PortType port []=
  {
        D4_GPIO_Port,D5_GPIO_Port,D6_GPIO_Port,D7_GPIO_Port
  };
  Lcd_PinType pin[] =
  {
        D4_Pin,D5_Pin,D6_Pin,D7_Pin,
  };

   lcd = Lcd_create(port, pin, RS_GPIO_Port, RS_Pin, EN_GPIO_Port, EN_Pin, LCD_4_BIT_MODE);

  MX_ADC1_Init();
  HAL_ADCEx_Calibration_Start(&hadc1);
  while (1)
  {
    HAL_ADC_Start(&hadc1);
    adc_value = HAL_ADC_GetValue(&hadc1);
    HAL_Delay(1);
    Lcd_cursor (&lcd,0,0);
    Lcd_string(&lcd, "Value:");
    Lcd_cursor(&lcd, 0, 6);
    Lcd_int(&lcd, adc_value);
    HAL_Delay(10);


  }
}
```

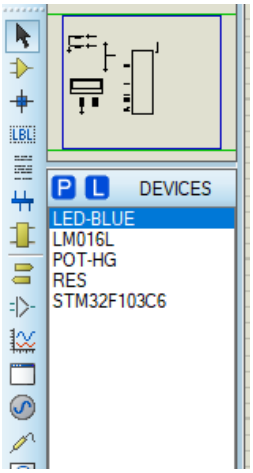
Gambar 5.7 Source Code Bab 5

✓ Lakukan proses build, pilih menu “Project – Build Project”

✓ Perhatikan hasil build (error, warning dan file .hex)

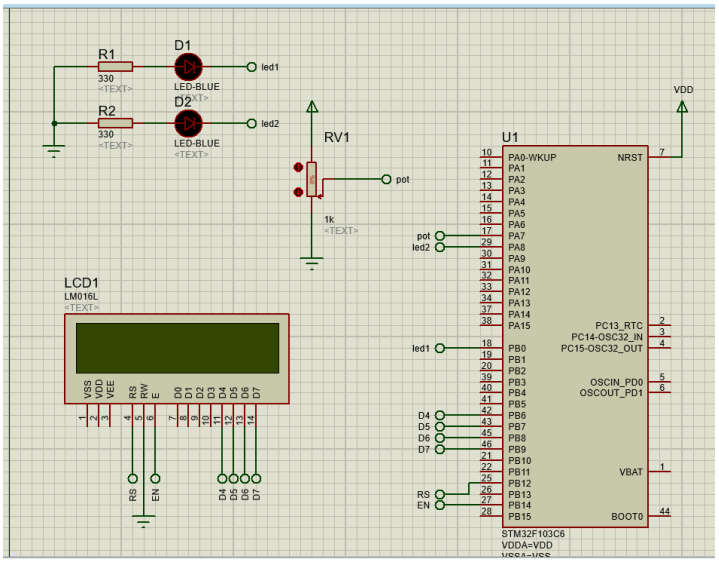
5.3.2. Langkah Percobaan Proteus

- ✓ Membuat project baru dengan nama “Bab 5” sesuai dengan materi pendahuluan di atas
- ✓ Setelah selesai membuat project baru dan berada pada tampilan simulasi silanjutnya silahkan memilih Component Mode – Pick From Library
- ✓ Maka akan muncul seperti gambar silahkan isi keywords dengan mengetik “POT_HG”,”RES”,”LM016L” lalu “Double Click” setiap pemilihan item.



Gambar 5.8 Component list

- ✓ Setelah itu silahkan ditambahkan sesuai dengan komponen list yang diinginkan.
- ✓ Jika benar maka akan muncul komponen pada component list
- ✓ Selanjutnya buat lah rangkaian seperti gambar di bawah ini



Gambar 5.9 Rangkaian Bab 5

- ✓ Untuk penambahan alat ukur voltmeter maupun amperemeter berada dalam “virtual instrument mode” pilih DC Voltmeter

- ✓ Setelah selesai merangkai sesuai gambar diatas dapat langsung menjalankan proses simulasi untuk mengetahui hasilnya dengan cara pilih tombol “Run the Simulation”  terletak kiri bawah.
- ✓ Untuk mengganti nilai dari komponen dapat dengan meng-klik 2 pada gambar komponen tersebut.

5.4 Analisa Percobaan
5.5 Kesimpulan