MODUL 5

Pemrograman Mikrokontroler AVR dasar

I. Pendahuluan

Pada praktikum ini, anda akan mempelajari cara mengembangkan sebuah sistem menggunakan mikrokontroler AVR buatan Atmel menggunakan software CodeVisionAVR. Seperti pada umumnya mikrokontroler, program untuk mikrokontroler AVR ditulis menggunakan bahasa assembly. CodeVisionAVR merupakan software C-cross compiler, dimana program dapat ditulis menggunakan bahasa-C. Dengan menggunakan pemrograman bahasa-C diharapkan waktu disain (deleloping time) akan menjadi lebih singkat. Setelah program dalam bahasa-C ditulis dan dilakukan kompilasi tidak terdapat kesalahan (error) maka proses download dapat dilakukan. Mikrokontroler AVR mendukung sistem download secara ISP (In-System Programming).

II. Tujuan

Setelah menyelesaikan praktikum ini, yang anda peroleh adalah :

- dapat menjelaskan arsitektur umum dan keistimewaan dari mikrokontroler AVR ATmega8535.
- dapat menceritakan kembali alur pengembangan software pada mikrokontroler AVR menggunakan CodeVisionAVR.
- dapat menggunakan fungsi input/output pada mikrokontroler AVR.
- dapat menggunakan fungsi kontrol LCD pada mikrokontroler AVR.
- dapat mengerjakan tugas yang diberikan dengan bahasa-C.

III. Gambaran Disain

Anda mencoba membuat project baru pada CodeVisionAVR untuk memanfaatkan fungsi input dan output mikrokontroler AVR ATmega8535. Fungsi input dan output tersebut adalah saklar dipswitch sebagai input, LED sebagai output, dan LCD sebagai output. Pada tugas yang akan dibuat, anda akan mencoba membuat mesin penjual minuman yang disimulasikan oleh push-button, LED dan LCD. Kemampuan pemrograman menggunakan bahasa-C sangat diperlukan pada praktikum ini.

IV. Dasar Teori

4.1. Mikrokontroler AVR

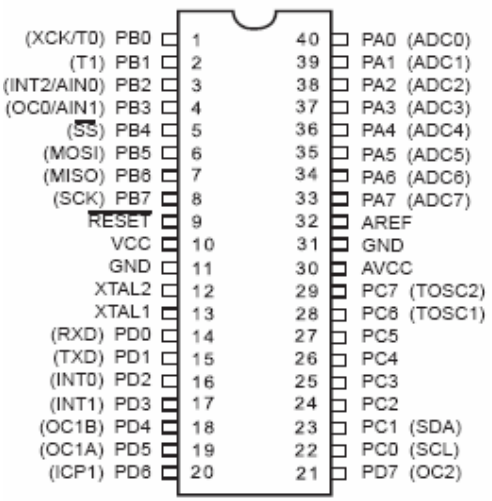
AVR merupakan seri mikrokontroler CMOS 8-bit buatan Atmel, berbasis arsitektur RISC (Reduced Instruction Set Computer). Hampir semua instruksi dieksekusi dalam satu siklus clock. AVR mempunyai 32 register general-purpose, timer/counter fleksibel dengan mode compare, interrupt internal dan eksternal, serial UART, programmable Watchdog Timer, dan mode power saving. Beberapa diantaranya mempunyai ADC dan PWM internal. AVR juga mempunyai In-System Programmable Flash on-chip yang mengijinkan memori program untuk diprogram ulang dalam sistem menggunakan hubungan serial SPI. Chip AVR yang digunakan untuk praktikum adalah ATmega8535.

ATmega8535 adalah mikrokontroler CMOS 8-bit daya-rendah berbasis arsitektur RISC yang ditingkatkan. Kebanyakan instruksi dikerjakan pada satu siklus clock, ATmega8535 mempunyai *throughput* mendekati 1 MIPS per MHz membuat disainer sistem untuk mengoptimasi komsumsi daya versus kecepatan proses.

Beberapa keistimewaan dari AVR ATmega8535 antara lain:

- Advanced RISC Architecture
    - 130 Powerful Instructions – Most Single Clock Cycle Execution
    - 32 x 8 General Purpose Working Registers
    - Fully Static Operation
    - Up to 16 MIPS Throughput at 16 MHz
    - On-chip 2-cycle Multiplier
- Nonvolatile Program and Data Memories
    - 8K Bytes of In-System Self-Programmable Flash
        - Endurance: 10,000 Write/Erase Cycles
    - Optional Boot Code Section with Independent Lock Bits
        - In-System Programming by On-chip Boot Program
        - True Read-While-Write Operation
    - 512 Bytes EEPROM
        - Endurance: 100,000 Write/Erase Cycles
    - 512 Bytes Internal SRAM
    - Programming Lock for Software Security
- Peripheral Features
    - Two 8-bit Timer/Counters with Separate Prescalers and Compare Modes
    - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
    - Real Time Counter with Separate Oscillator
    - Four PWM Channels
    - 8-channel, 10-bit ADC
        - 8 Single-ended Channels
        - 7 Differential Channels for TQFP Package Only
        - 2 Differential Channels with Programmable Gain at 1x, 10x, or 200x for TQFP Package Only
    - Byte-oriented Two-wire Serial Interface

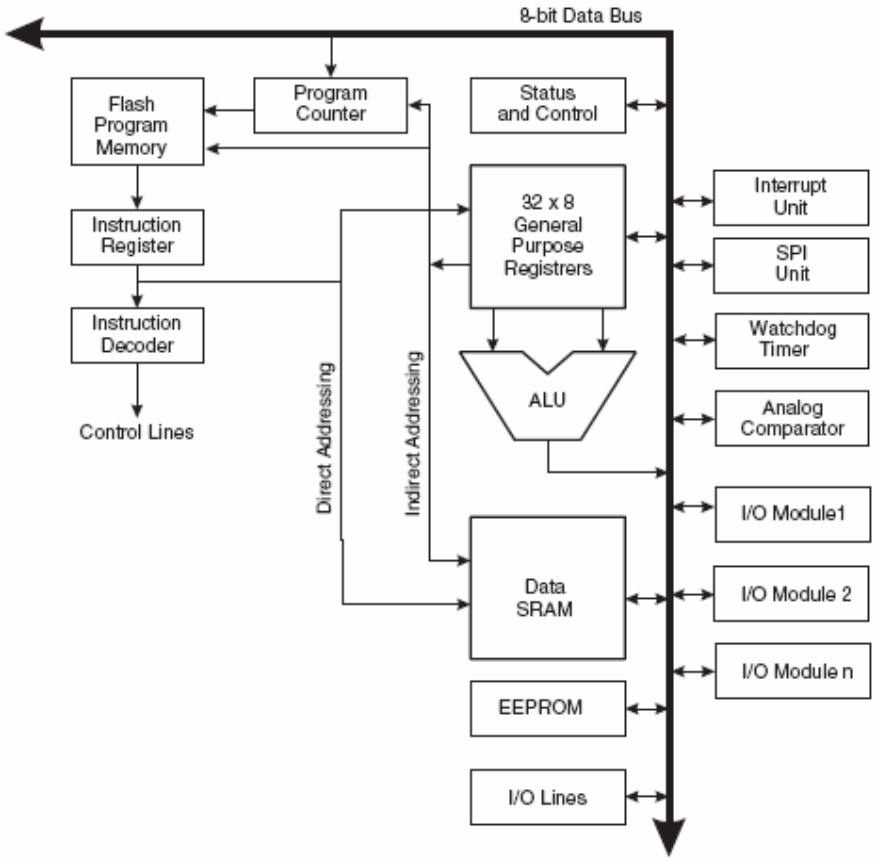
  - Programmable Serial USART
  - Master/Slave SPI Serial Interface
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
- Special Microcontroller Features
  - Power-on Reset and Programmable Brown-out Detection
  - Internal Calibrated RC Oscillator
  - External and Internal Interrupt Sources
  - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby and Extended Standby
- I/O and Packages
  - 32 Programmable I/O Lines
  - 40-pin PDIP, 44-lead TQFP, 44-lead PLCC, and 44-pad MLF
- Operating Voltages
  - 2.7 - 5.5V for ATmega8535L
  - 4.5 - 5.5V for ATmega8535
- Speed Grades
  - 0 - 8 MHz for ATmega8535L
  - 0 - 16 MHz for ATmega8535



Gambar 1. Pin-pin ATmega8535 kemasan 40-pin

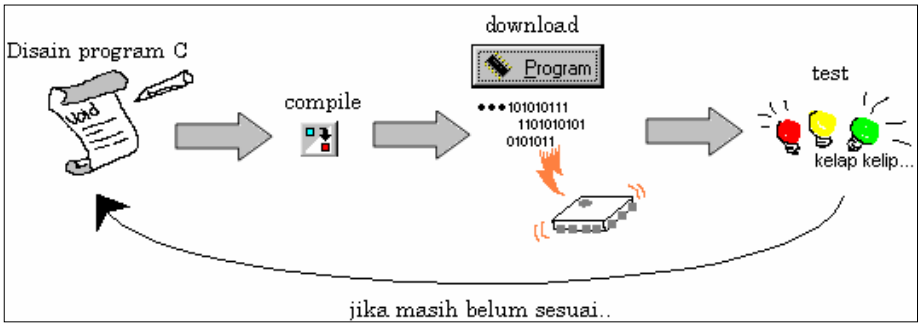
Pin-pin pada ATmega8535 dengan kemasan 40-pin DIP (dual in-line package) ditunjukkan oleh gambar 1.

Guna memaksimalkan performa dan paralelisme, AVR menggunakan arsitektur Harvard (dengan memori dan bus terpisah untuk program dan data). Arsitektur CPU dari AVR ditunjukkan oleh gambar 2. Instruksi pada memori program dieksekusi dengan pipelining single level. Selagi sebuah instruksi sedang dikerjakan, instruksi berikutnya diambil dari memori program.



Gambar 2. Arsitektur CPU dari AVR

Program ditulis menggunakan tool CodeVisionAVR. CodeVisionAVR merupakan cross-compiler. Program cukup ditulis menggunakan bahasa-C.



Gambar 3. Alur pemrograman AVR menggunakan CodeVisionAVR

4.2. Port sebagai input/output digital

ATmega8535 mempunyai empat buah port yang bernama PortA, PortB, PortC, dan PortD. Keempat port tersebut merupakan jalur bi-directional dengan pilihan internal pull-up.

Tiap port mempunyai tiga buah register bit, yaitu DDxn, PORTxn, dan PINxn. Huruf ‘x’ mewakili nama huruf dari port sedangkan huruf ‘n’ mewakili nomor bit. Bit DDxn terdapat

pada I/O address DDRx, bit PORTxn terdapat pada I/O address PORTx, dan bit PINxn terdapat pada I/O address PINx.

Bit DDxn dalam regiter DDRx (Data Direction Register) menentukan arah pin. Bila DDxn diset 1 maka Px berfungsi sebagai pin output. Bila DDxn diset 0 maka Px berfungsi sebagai pin input.

Bila PORTxn diset 1 pada saat pin terkonfigurasi sebagai pin input, maka resistor pull-up akan diaktifkan. Untuk mematikan resistor pull-up, PORTxn harus diset 0 atau pin dikonfigurasi sebagai pin output. Pin port adalah tri-state setelah kondisi reset.

Bila PORTxn diset 1 pada saat pin terkonfigurasi sebagai pin output maka pin port akan berlogika 1. Dan bila PORTxn diset 0 pada saat pin terkonfigurasi sebagai pin output maka pin port akan berlogika 0.

Saat mengubah kondisi port dari kondisi *tri-state* (DDxn=0, PORTxn=0) ke kondisi *output high* (DDxn=1, PORTxn=1) maka harus ada kondisi peralihan apakah itu kondisi *pull-up enabled* (DDxn=0, PORTxn=1)atau kondisi *output low* (DDxn=1, PORTxn=0). Biasanya, kondisi pull-up enabled dapat diterima sepenuhnya, selama lingkungan impedansi tinggi tidak memperhatikan perbedaan antara sebuah *strong high driver* dengan sebuah pull-up. Jika ini bukan suatu masalah, maka bit PUD pada register SFIOR dapat diset 1 untuk mematikan semua pull-up dalam semua port.

Peralihan dari kondisi *input dengan pull-up* ke kondisi *output low* juga menimbulkan masalah yang sama. Kita harus menggunakan kondisi tri-state (DDxn=0, PORTxn=0) atau kondisi output high (DDxn=1, PORTxn=0) sebagai kondisi transisi.

Lebih detil mengenai port ini dapat dilihat pada manual datasheet dari IC ATmega8535.

Tabel 1. Konfigurasi pin port

DDxn	PORTxn	PUD (in SFIOR)	I/O	Pull-up	Comment
0	0	X	Input	No	Tri-state (Hi-Z)
0	1	0	Input	Yes	Pxn will source current if ext. pulled low.
0	1	1	Input	No	Tri-state (Hi-Z)
1	0	X	Output	No	Output Low (Sink)
1	1	X	Output	No	Output High (Source)

Bit	7	6	5	4	3	2	1	0	
	ADTS2	ADTS1	ADTS0	–	ACME	PUD	PSR2	PSR10	SFIOR
Read/Write	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Bit 2 – PUD : Pull-up Disable
Bila bit diset bernilai 1 maka pull-up pada port I/O akan dimatikan walaupun register DDxn dan PORTxn dikonfigurasikan untuk menyalakan pull-up (DDxn=0, PORTxn=1).

4.3. Rutin-rutin standar

Pada software CodeVisionAVR telah disediakan beberapa rutin standar yang dapat langsung digunakan. Anda dapat melihat lebih detil pada manual dari CodeVisionAVR. Beberapa contoh fungsi yang telah disediakan antara lain adalah:

Fungsi LCD
Berada pada header **lcd.h** yang harus di-*include*-kan sebelum digunakan. Sebelum melakukan *include* terlebih dahulu disebutkan pada port mana LCD akan diletakkan. Hal ini juga dapat dengan mudah dilakukan dengan menggunakan CodeWizardAVR.

```
/* modul LCD dihubungkan dengan PORTC */
#asm
    .equ __lcd_port=0x15
#endasm

/* sekarang fungsi LCD dapat di-include*/
#include <lcd.h>
```

Fungsi-fungsi untuk mengakses LCD diantaranya adalah :

- **unsigned char lcd_init(unsigned char lcd_columns)**
  Untuk menginisialisasi modul LCD, menghapus layar dan meletakkan posisi karakter pada baris ke-0 kolom ke-0. Jumlah kolom pada LCD harus disebutkan (misal, 16). Kursor tidak ditampakkan. Nilai yang dikembalikan adalah 1 bila modul LCD terdeteksi, dan bernilai 0 bila tidak terdapat modul LCD. Fungsi ini harus dipanggil pertama kali sebelum menggunakan fungsi yang lain.

- **void lcd_clear(void)**
  Menghapus layar LCD dan meletakkan posisi karakter pada baris ke-0 kolom ke-0.

- **void lcd_gotoxy(unsigned char x, unsigned char y)**
  Meletakkan posisi karakter pada kolom ke-x baris ke-y. Nomor baris dan kolom dimulai dari nol.

- **void lcd_putchar(char c)**
  Menampilkan karakter c pada LCD.

- **void lcd_puts(char *str)**
  Menampilkan string yang disimpan pada SRAM pada LCD.

Fungsi Delay
Menghasilkan delay dalam program-C. Berada pada header **delay.h** yang harus di-*include*-kan sebelum digunakan. Sebelum memanggil fungsi, interrupsi harus dimatikan terlebih dahulu, bila tidak maka delay akan lebih lama dari yang diharapkan. Juga sangat penting untuk menyebutkan frekuensi clock chip IC AVR yang digunakan pada menu Project-Configure-C Compiler-Code Generation.

Fungsi delay yang disediakan adalah:

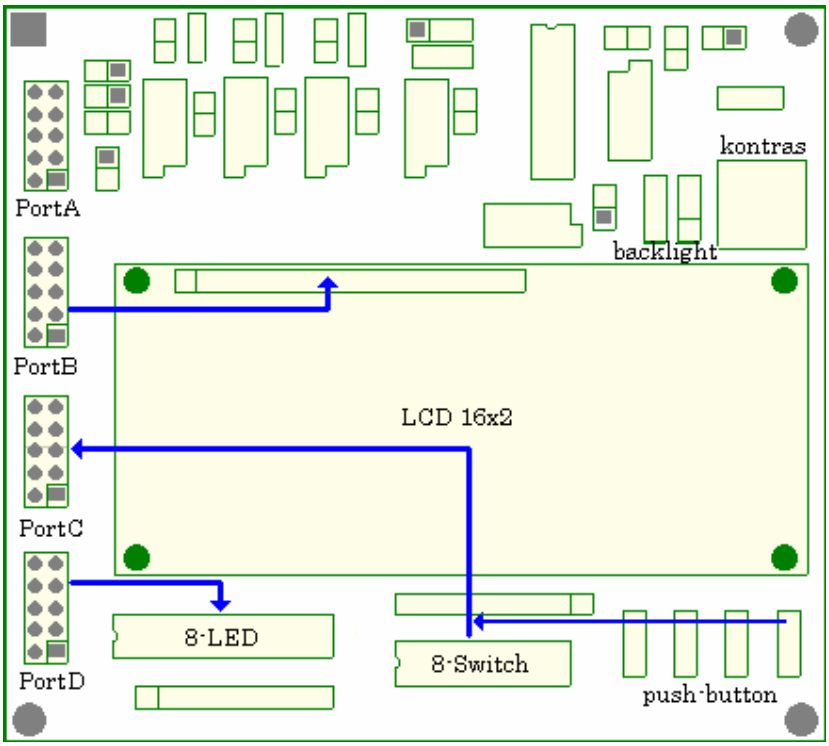
- **void delay_us(unsigned int n)**
  menghasilkan delay selama n μ-detik, n adalah nilai konstan

- **void delay_ms(unsigned int n)**
  menghasilkan delay selama n mili-detik, n adalah nilai konstan

Kedua fungsi tersebut secara otomatis akan me-reset watchdog-timer setiap 1 milidetik dengan mengaktifkan instruksi **wdr**.

4.4. Modul I/O

Modul I/O yang digunakan mempunyai skema tampak atas seperti pada gambar 4. Sedangkan hubungan port AVR dengan I/O yang digunakan ditunjukkan oleh tabel 2.



Gambar 4. Modul I/O AVR yang digunakan

Tabel 2. Hubungan port AVR dengan I/O

Port B		Port C			Port D	
Bit ke-	LCD (pin)	Bit ke-	Switch nomor-*	Push button ke-*	Bit ke-	LED ke-*
0	RS (4)	0	8	4	0	10
1	R/W (5)	1	7	3	1	9
2	EN (6)	2	6	2	2	8
3	-	3	5	1	3	7
4	DB4 (11)	4	4	-	4	6
5	DB5 (12)	5	3	-	5	5
6	DB6 (13)	6	2	-	6	4
7	DB7 (14)	7	1	-	7	3

*) nomor dihitung dari kiri ke kanan

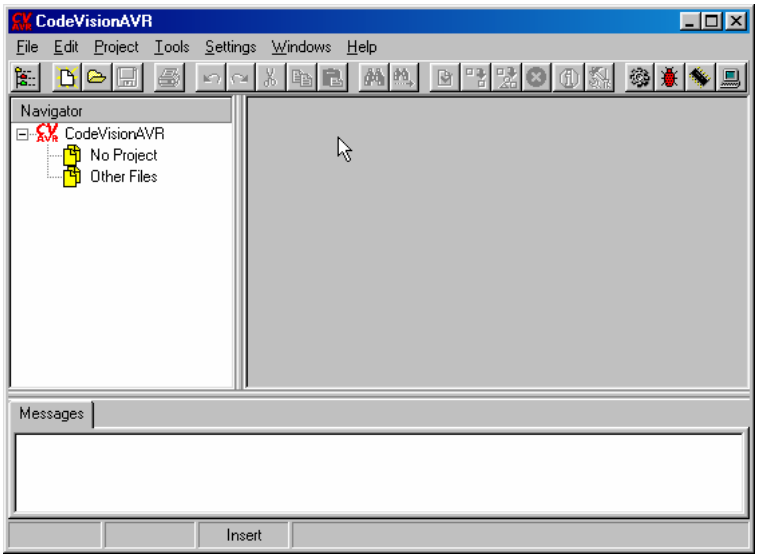
V. Peralatan

- 1 set PC yang dilengkapi dengan software CodeVision AVR.
- 1 set development board AVR ATmega8535
- 1 power-supply +9VDC

VI. Prosedur Praktikum

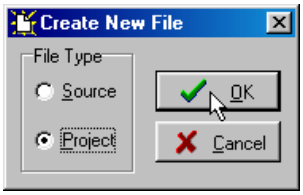
Memulai project baru

1. Jalankan software CodeVisionAVR.



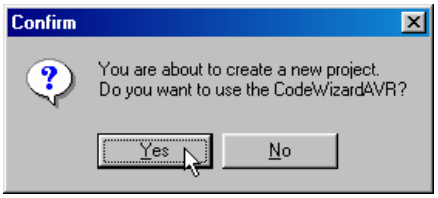
Gambar 4. CodeVisionAVR

2. Buatlah project baru. Pilih **File → New**. Pilih project lalu tekan tombol OK.



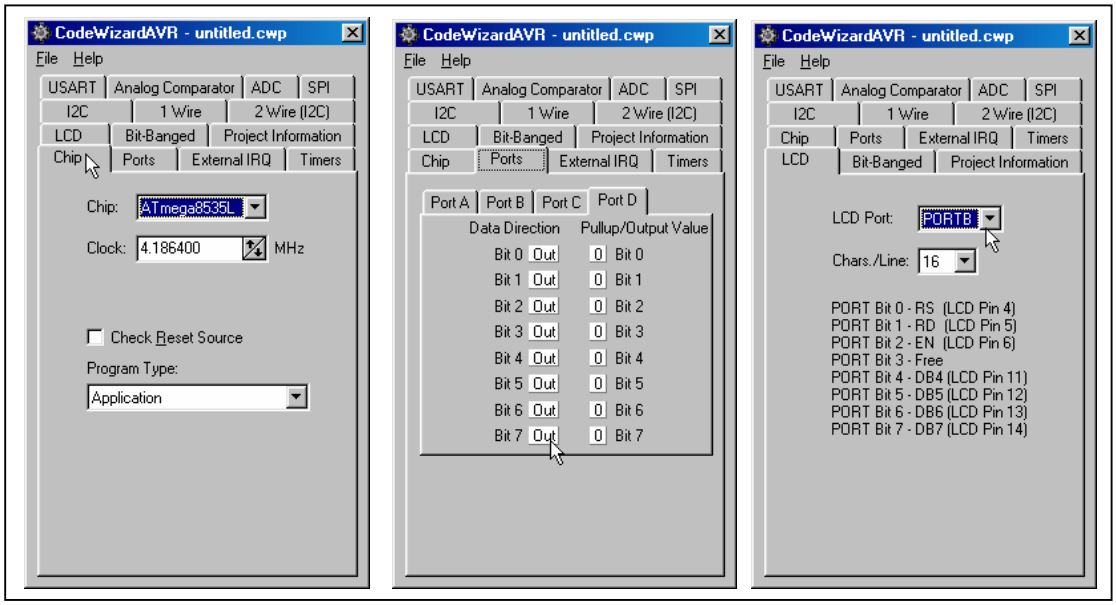
Gambar 5. Create project baru

3. Kemudian muncul dialog apakah akan menggunakan CodeWizardAVR untuk mempermudah merancang kerangka program. Pilih YES.



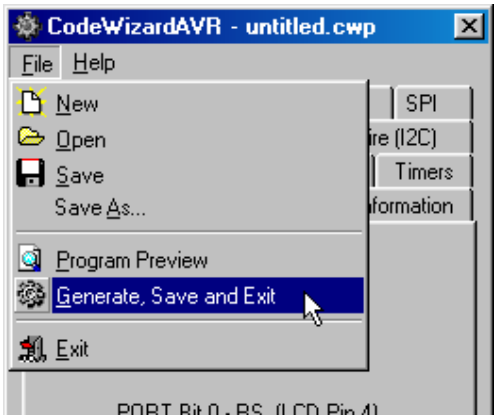
Gambar 6. Pilihan untuk menggunakan CodeWizardAVR

4. Board yang digunakan menggunakan chip ATmega8535 dengan clock 4MHz, kemudian pilih tab Ports, secara default port merupakan pin input, lakukan seting untuk portD sebagai output, lalu pilih tab LCD, gunakan portB karena pada board LCD terhubung dengan portB.



Gambar 7. Menseting jenis chip, port I/O dan port untuk LCD

5. Kemudian pilih **File → Generate, Save and Exit**.



Gambar 8. Menghasilkan kode program

6. Simpan pada direktori **e:\embedded**. Beri nama source file, nama project file, dan nama codewizard project file anda.

7. Pada header program tambahkan include berikut

```
#include <delay.h>
```

8. Pada main program, sisipkan perintah berikut

```
...
// LCD module initialization
lcd_init(16);
lcd_gotoxy(0,0);
lcd_putsf("Tekan pushbutton");

while (1)
      {
      // Place your code here
         if (PINC & 1 == 1){
          lcd_gotoxy(0,0);
          lcd_putsf("Tekan pushbutton");
          PORTD = 0x00;
          delay_ms(200);
         }else{
          lcd_gotoxy(0,0);
          lcd_putsf("PB1 ditekan !   ");
          PORTD = 0xFF;
          delay_ms(200);
         }
      };
}
```

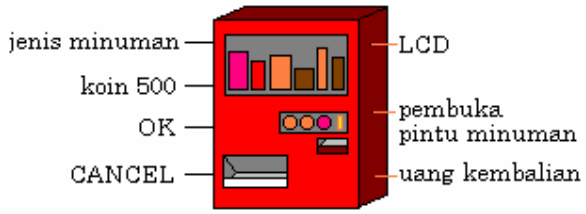
9. Sebelum melakukan kompilasi, lakukan seting berikut untuk pertama kali saja. Pilih **Project → Configure**. Pastikan pada tab **C Compiler** nama *chip* dan *clock* telah sesuai dengan IC yang terpasang pada board. Dan pada tab **After Make** pilih *Program the Chip*. Lalu tekan tombol OK.
   Pilih **Settings → Programmer**, pilih *Kanda System STK200+/300* untuk AVR Chip Programmer Type. Lalu tekan tombol OK.

10. Secara bertahap anda dapat melakukan check syntax, compile, dan make. Atau langsung pilih **Project → Make**. Bila tidak ada error maka akan muncul kotak dialog informasi. Siapkan board, nyalakan suplai tegangannya, hubungkan kabel download dari PC ke board. Kemudian tekan tombol **Program**.

11. Bila tidak ada kesalahan maka pada layar LCD akan ditampilkan tulisan "Tekan pushbutton". Cobalah menekan pushbutton nomor 4 (sebelah kanan), maka pada layar LCD akan ditampilkan tulisan "PB1 ditekan" dan LED menyala. Apabila push-button dilepas maka tampilan layar LCD kembali seperti semula dan LED padam.

VII. Laporan sementara

Tugas berikut dikerjakan sebagai laporan sementara.
Buatlah miniatur mesin penjual minuman dengan ketentuan sebagai berikut:



<u>4 - tombol input :</u>
* memilih jenis minuman
* memasukkan koin
* tombol OK
* tombol CANCEL

<u>Output :</u>
* tampilan LCD
* pembuka pintu minuman
* uang kembalian

1. Jenis minuman dan harga : teh (Rp. 1000), kopi ginseng (Rp 1500), coklat dingin (Rp. 2000).
2. Kondisi pembuka pintu minuman diwakili oleh led: merah (tertutup), hijau (terbuka).
3. Kondisi uang kembalian diwakili oleh led oranye: bila led menyala = ada kembalian.
4. User memilih minuman yang diinginkan, nama minuman ditampilkan pada baris pertama LCD.
5. User memasukkan uang logam 500, jumlah uang yang dimasukkan ditampilkan pada baris kedua LCD.
6. Bila user menekan tombol OK maka:
    a. Bila uang kurang maka diinformasikan pada LCD, semua uang dikembalikan
    b. Bila uang pas maka diinformasikan pada LCD, pintu minuman terbuka
    c. Bila uang lebih maka diinformasikan pada LCD, pintu minuman terbuka dan uang kembalian dikeluarkan
7. Bila user menekan tombol CANCEL maka diinformasikan pada LCD, semua uang yang dimasukkan akan dikembalikan.

VIII. Laporan resmi

Jawablah pertanyaan berikut untuk bagian analisa pada laporan resmi:

1. Mendisain program menggunakan bahasa-C memberikan kemudahan bagi programer. Menurut anda perlukah kita mengetahui (belajar) bahasa assemblernya? Mengapa?
2. Buatlah flowchart dari tugas yang anda kerjakan!

IX. Tambahan

Berikan saran atau komentar guna pengembangan lebih lanjut praktikum ini.