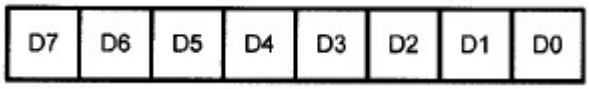
Modul Pendamping 1

Pemrograman I/O (Input/Ouput) Mikrokontroler dengan Bahasa Assembly

Pada awal masa komputer, programmer mengembangkan Bahasa Assembly sebagai pengembangan dari "bahasa mesin" yang terdiri dari angka-angka 0 dan 1 disertai dengan *mnemonic* yang merupakan kode/singkatan yang mudah untuk diingat. Bahasa ini kemudian diterjemahkan sebagai Bahasa mesin dengan menggunakan program "assembler". Bahasa Assembly merupakan *low-level language* karena berhubungan langsung dengan struktur internal CPU.

CPU menggunakan register untuk menyimpan data secara sementara. Pemahaman tentang register dan arsitektur dari CPU yang digunakan adalah hal yang penting untuk melakukan pemrograman dalam Bahasa Assembly. Register-register pada AVR merupakan register-register 8-bit dengan rentang dari D7 yang merupakan MSB (*most significant bit*) hingga D0 yang merupakan LSB *(least significant bit*), seperti ditunjukkan pada Gambar 1. AVR memiliki 32 register serbaguna yang terdiri dari R0-R31 yang dapat digunakan untuk melakukan tugas aritmatika ataupun logika. Beberapa perintah yang digunakan dalam Bahasa Assembly diantaranya adalah `LDI` (*Load Immediate*), `OUT`, `IN`, `DEC` (*Decrement*), `ADIW` (*Add Immediate to Word*), dan `COM` (*One's Complement*).



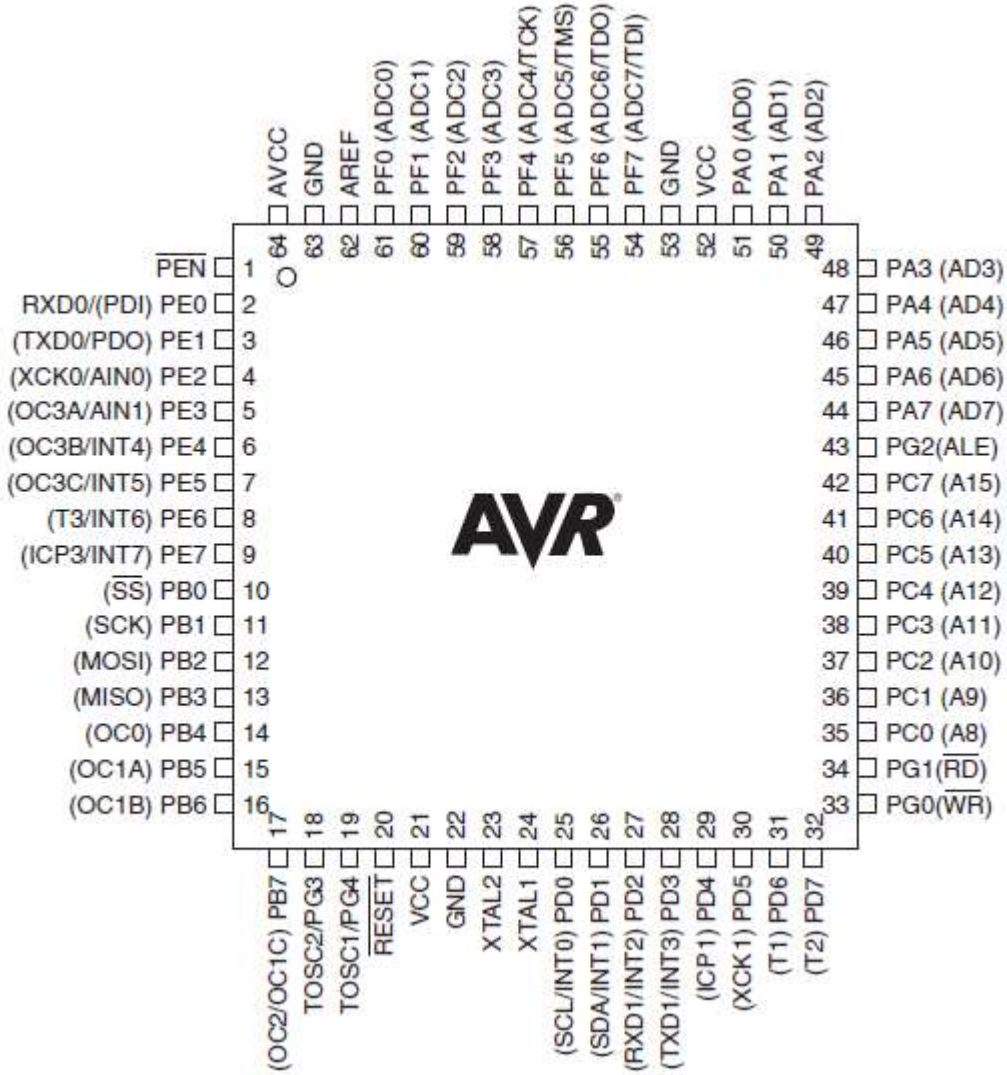
Gambar 1 Register 8 bit.
SUMBER: MA MAZIDI

Representasi format data untuk bilangan *hex* dapat dilakukan dengan menuliskan 0x atau $ di depan angka (e.g. 0x18 atau $18). Bilangan biner dituliskan dengan menuliskan 0b diikuti dengan bilangan binernya (e.g. 0b010101), sedangkan bilangan decimal dituliskan dengan mengunakan angka desimalnya saja (e.g. 21).

AVR dapat melakukan proses *looping*, yaitu mengulang urutan perintah atau operasi sebanyak beberapa kali. Salah satu cara untuk melakukan proses *looping* adalah dengan menggunakan perintah `BRNE` (*Branch if Not Equal*). Selain itu, terdapat beberapa cara untuk melakukan control instruksi transfer, seperti:

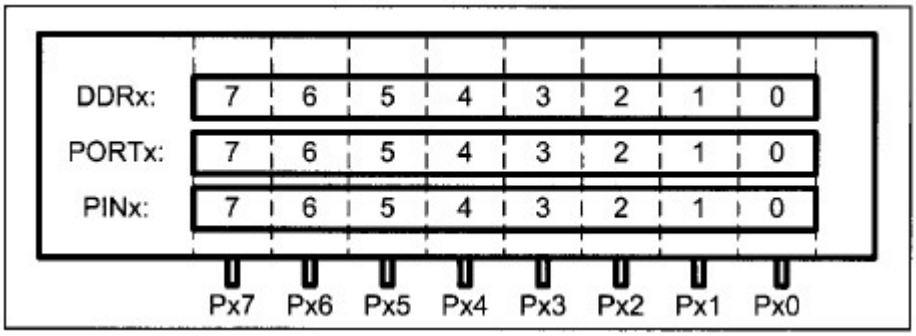
1. Melakukan lompatan tanpa syarat (*unconditional jump*) yang dapat mengendalikan lokasi target, salah satunya dengan perintah RJMP (*Relative Jump*).
2. Memanggil *subroutine* untuk melakukan perintah-perintah yang sering digunakan dengan perintah `RCALL` (*Relative Call to Subroutine*) serta perintah `RET` (*Return from Subroutine*) pada akhir *subroutine*.

Pada keluarga AVR, terdapat banyak port untuk operasi I/O, seperti PORTA, PORTB, hingga PORTF (8-bit), serta PORTG (5-bit), yang terdapat pada ATmega128 seperti ditunjukkan pada Gambar 2. Port-port ini dapat digunakan jika terlebih dahulu diprogram. Masing-masing port juga memiliki fungsi-fungsi lainnya seperti ADC, *timer*, *interrupts*, dan komukasi serial.



Gambar 2 Pinout ATmega128
SUMBER: ATMEL

Setiap port didesain sebagai PORTx, DDRx (*Data Direction Register*), dan PINx (*Port INput*) dengan register 8-bit dan 8 jumlah pin maksimum, seperti ditunjukkan pada Gambar 3. Register I/O DDRx menentukan sebuah port menjadi input atau output. Register PINx membaca data pada pin, sedangkan register PORTx digunakan untuk mengirim data.



Gambar 3 Hubungan antara regiser dengan pin pada AVR
SUMBER: MA MAZIDI

Referensi:

1. Atmel, “8-bit Atmel Microcontroller with 128Kbytes In-System Programmable Flash” ATmega128 and ATmega128L datasheet, Rev. 2467X-AVR-06/11.
2. Atmel, “Atmel AVR 8-bit Instruction Set” Instruction Set Manual, Rev. 0856K–AVR–05/2016.
3. Muhammad Ali Mazidi, Sarmad Naimi, and Sepehr Naimi. 2011. *AVR Microcontroller and Embedded Systems: Using Assembly and C*. Prentice Hall Press, Upper Saddle River, NJ, USA.