PERANCANGAN CONTROLLER

Djunaidi Santoso

Jurusan Teknik Informatika, Fakultas Ilmu Komputer, Bina Nusantara University
Jln. K. H. Syahdan No 9, Palmerah, Jakarta Barat 11480
djunsan@yahoo.com

ABSTRACT

In making controller, there must be knowledge about component controlling functions that used for controller designing. In the control unit, it needs simple method as micro programming. This micro programming is to create a micro program in binary numbering that used for controlling pin's component controller and outside the controller. Controller design in general needs several prerequisites, which are digital system and controller and assembly language.

Keywords: *controller creating, controller component, control unit, micro programming, binary numbering pin's controlling.*

ABSTRAK

Dalam melakukan pembuatan controller perlu adanya pengetahuan mengenai fungsi komponen-komponen controller yang digunakan untuk perancangan controller tersebut. Dalam hal ini untuk kontrol unitnya diperlukan metoda yang sederhana yaitu pemrograman mikro. Pemrograman mikro yang dipakai untuk membuat program mikro dalam angka biner yang dipakai untuk pengendalian pin's komponen controller maupun luar controller. Perancangan controller secara umum ini perlu prasyarat yaitu: sistem digital dan controller dan assembly language.

Kata kunci: *pembuatan controller, komponen controller, control unit, pemrograman mikro, angka biner pengendali pin's.*

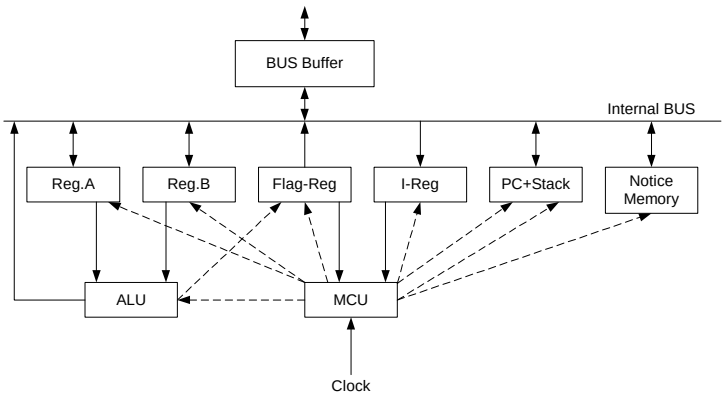
PENDAHULUAN

Teknik pemrograman mikro adalah sangat penting dalam merancang central processing unit dan khususnya dalam hal pembuatan instruksi-instruksi bahasa mesin/rakitan. Teknik ini masih banyak dipakai atau dipergunakan disetiap perguruan tinggi di indonesia dalam bidang perancangan *controller*. Penelitian ini untuk mempermudah atau mempercepat pengertian, apa yang dimaksud dengan pemrograman mikro? Perlu juga dijelaskan pembahasan masalah komponen *controller*, cara pemrosesan *controller*, arsitektur *controller* secara umum dan membahas masalah prinsip pemrograman mikro dengan suatu contoh yang dapat langsung diaplikasikan. *Controller* dengan unit kontrol adalah penerusan yang berdasarkan contoh sebelumnya.

Landasan Teori

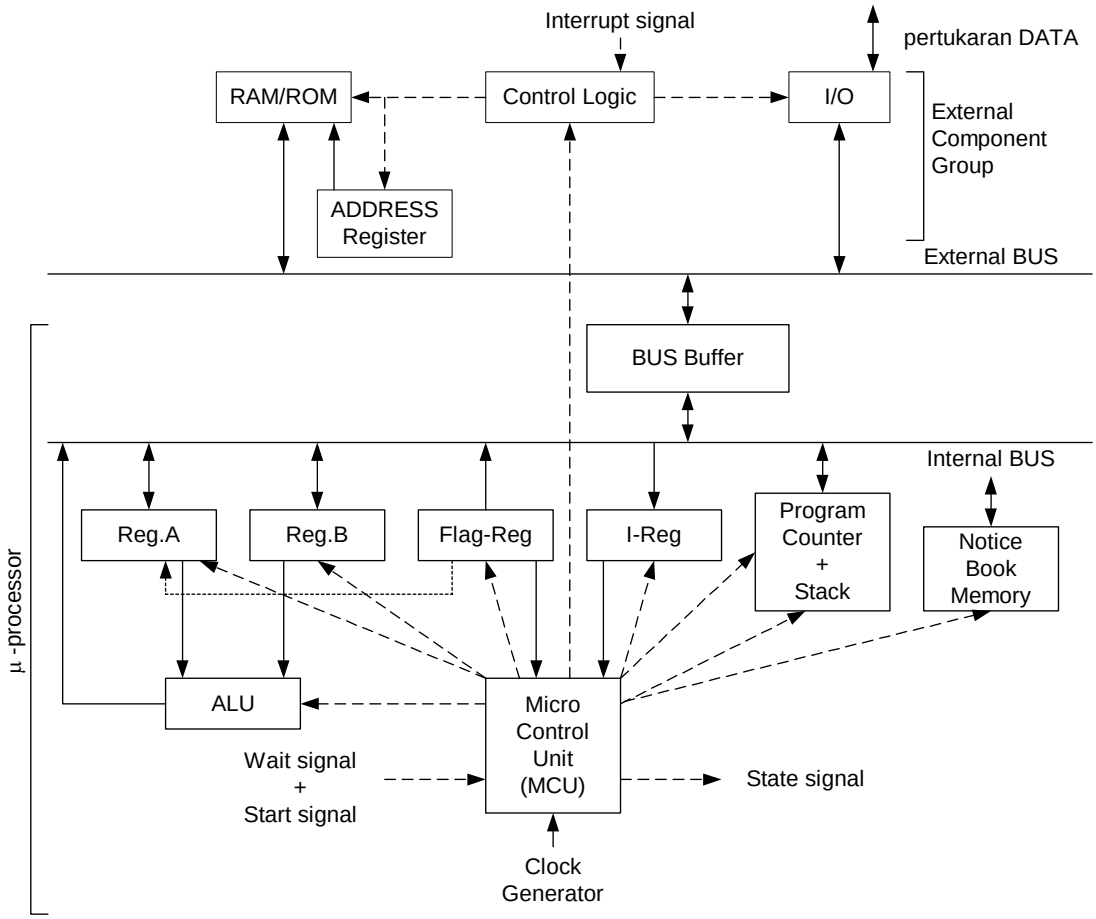
Komponen-komponen *controller* merupakan fungsi dari beberapa register *controller* perlu diketahui dalam hal merancang *controller*. Pengenalan komponen *controller* secara umum dan cara kerjanya ialah (1) program *counter-register*, suatu register pencacah alamat program yang berada pada RAM (*Random Access Memory*); (2) *general register*, sebuah register umum untuk dipergunakan sebagai tempat menyimpan informasi seperti AX-Register, BX-Register, CX-Register, DX-Register atau Register A, Register B dan lain-lain; (3) register instruksi (*instriction register*), register tempat menampung kode instruksi, yang berasal dari memori melalui Data-Bus ke Instruction-register yang disebut proses *fetching*; (4) register bendera (flag-register), register untuk menampung sinyal hasil dari pemrosesan Aritmetik dan logik serta lainnya seperti pemrosesan penambahan dapat menghasilkan suatu *carry*, pemrosesan pengurangan dapat menghasilkan suatu *borrow* atau negatif, pemrosesan perkalian akan dapat menghasilkan suatu hasil yang melebihi dari register yang sudah disediakan—perkalian 8 bits hasilnya bisa melebihi dari jumlah bits yang telah dibuat oleh perusahaan yang membuat register, jika ini terjadi bendera overflow = 1; (5) *stack register*, tempat untuk menyimpan alamat kembali, jika ada instruksi call atau untuk menyimpan/melindungi data dari register umum, jika ada instruksi PUSH. Jelas Stack register kapasitasnya lebih kecil daripada Stack memory; (6) *index register*, register untuk menyimpan hasil dari manipulasi alamat-memori utama/real/internal. Pada INTEL 8088 register ini diganti dengan ALU (Arithmetic Logic Unit) dengan tanda SIGMA; (7) *control unit*, suatu unit (terdiri dari banyak rangkaian) untuk mengontrol PINs komponen rangkaian digital yang disesuaikan dengan kode instruksi yang pada waktu itu sedang dikerjakan.

Gambar Umum Blok *Controller*



Gambar 1 Gambaran umum blok *controller*

Karena komponen itu tidak menggambarkan suatu komputer digital, maka disebut *controller*. Dan dapat dinamakan *controller*. Ada suatu pertanyaan, kenapa tidak dipergunakan semaksimal mungkin tempat-tempat yang masih kosong pada ROM/RAM sebagai instruction register? Untuk menjawab pertanyaan tersebut, harus dilihat pemrosesan program secara keseluruhan, bahwa seluruh pentransportasian informasi melalui BUS, kecuali control information (dari MCU). BUS ini memerlukan clock untuk setiap pentransferan informasi lainnya. Oleh karena itu setelah kejadian pentransferan informasi, BUS segera dikosongkan, dimana dibiarkan penyimpanan kondisi-kondisi (yang ditransfer) langsung pada si penerima. Setiap tempat pengirim dan tempat penerima harus mempunyai register atau memori lainnya. Gambar 1 menjelaskan tentang minimal konfigurasi yang dapat mengendalikan suatu sistem.



Gambar 2 Minimal konfigurasi yang dapat mengendalikan suatu sistem.

Begitupun Memori Catatan (*Notice-Book-Memory*) yang terdiri dari register-register mempunyai keuntungan-keuntungan, bahwa alamatnya pendek, sehingga menghasilkan sedikit Memory Access Time, jika bekerja dengannya dibandingkan dengan menggunakan ROM/RAM. Atasa dasar yang sama (mengenai keuntungan waktu), tercapai juga fungsi dari Stack memory, yang registernya sebagai address memory untuk suatu CALL-SUBROUTINE.
Flag register terdiri dari bit-bit tersendiri, yang diset atau tidak disetting dan ini tergantung dari hasil ALU/lainnya.

Penambahan Control Logic

Penambahan *control logic* harus didekodekan untuk pengaturan ROM (RAM), *Address-Register*, I/O yang dikendalikan *micro control unit*.

Penelitian *Controller* Sederhana

Penelitian *controller* sederhana akan membuat 16 intruksi dasar, yaitu instruksi transport, instruksi aritmetik, instruksi percabangan (*go to*/ *if then*), dan bus yang mempunyai 8 saluran parallel. Setiap data dan *address-word* mempunyai 8 bits = 1 byte, sehingga saluran Bus dipakai secara maksimal. Dengan 8 bits lebar wordnya dapat menghasilkan 256 macam word dan dapat mengalamatkan suatu memory external sebanyak 256 sel. Berarti juga 256 instruksi dapat tertampung pada memori. Mikroprosesor ini mempunyai 3 register sebagai notes book memory (register umum) yaitu: Reg. A, Reg. B, Reg. C. Reg. A disebut Akumulator. Reg. A dan Reg. B sekaligus membentuk *Operand-Memory* untuk Aritmetik Logic Unit (ALU).

Selain dari komponen tersebut di atas, *controller* harus memiliki *program counter* (8 bits), *stack register* (8 bits), register penangkap informasi (*buffer register* = 8 bits), *flag register* (2 Bits untuk *zero* bit dan *carry* bit), *instruction register* (IR=4 bits), *Micro Control Unit* (MCU), register A, register B serta register C (8 bits).

Analisa Prinsip Mikroprogram

Mengendalikan pin kontrol sesuai dengan instruksi yang diciptakan. Dalam memori tidak hanya data dan hasil sementara saja, melainkan juga angka-angka serta *Control Information* yaitu program, yang terdiri dari insruksi-instruksi. Rangkaian dari percobaan ini dibuat sedemikian, bahwa pengambilan instruksi-instruksi dan penyimpanan sementara dari instruksi-instruksi pada *intruction register* dapat dimungkinkannya.

Persiapan

Persiapan yang dilakukan adalah dengan mendapatkan informasi mengenai kata-kata: (1) instruksi; (2) bagian operasi (*operation path*) = 3 bits *operation code* bagian dari alamat (*address-path*) = 5 bits. Semuanya 8 bits; (3) *instruction register*; (4) *program counter* (*instruction counter*); (5) *instruction cycles*; (6) *instruction holdphase*; (7) *instruction execution phase*; (8) *tristate-gate*; (9) *control signal*; (10) *microprogram*.

Rangkaian

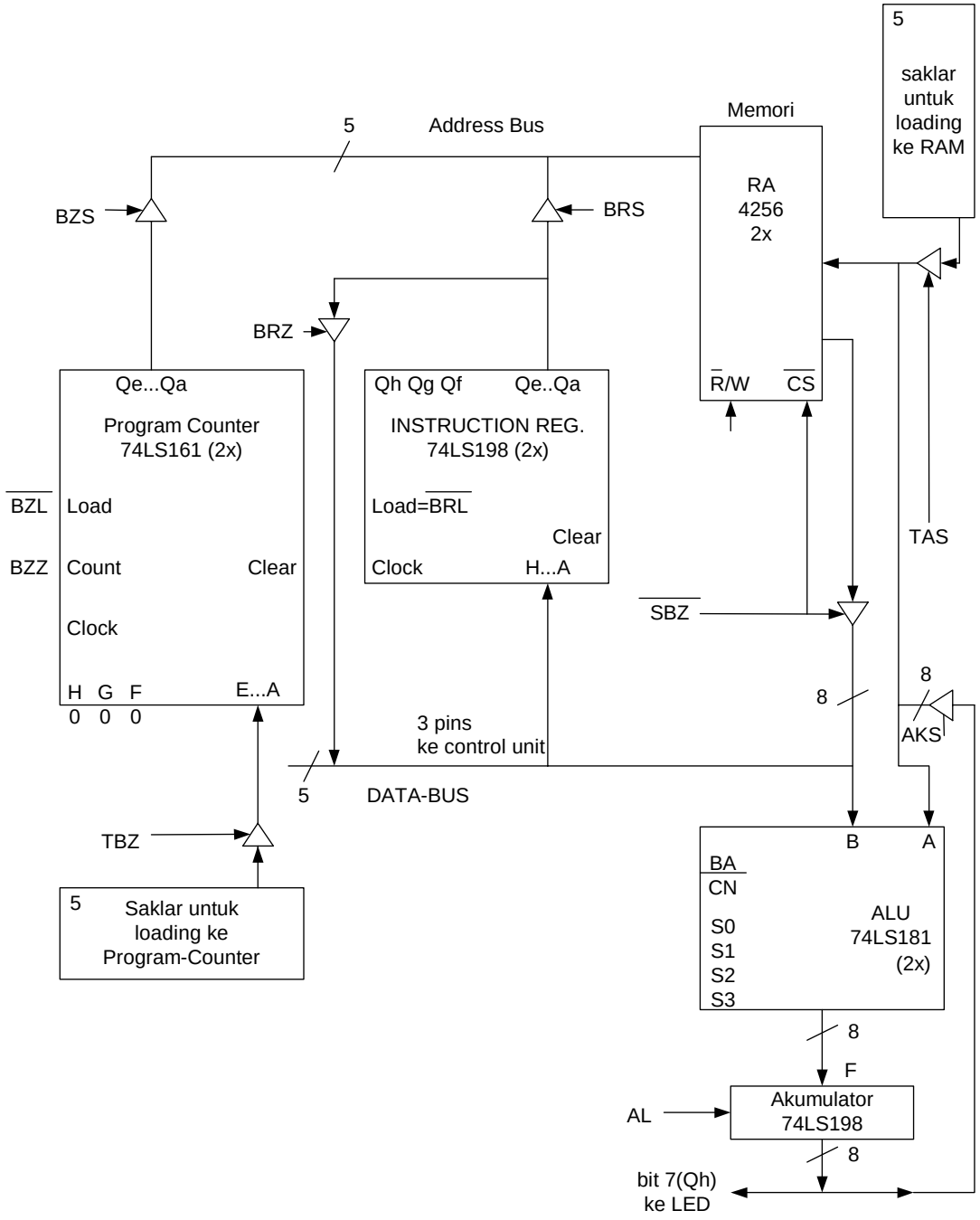
Pembuatan rangkaian computer kecil ini yaitu: (1) 8 bits (*word length*), untuk instruction words, operation code = 3 bits, address path = 5 bits; (2) outputs dari program counter melalui tristate gate ke Address-Bus; (3) *address field* dari *instruction register* di hubungkan ke *Address-Bus* melalui *Tristate-Gate*, juga kaki masukkan *Program-Counter* dan *Instruction Register* diletakkan pada *Data-Bus*; (4) akhirnya dibuat suatu hubungan dari accumulator ke *Memori-Input*; (5) Untuk masukkan ke memori di gunakan saklar memlalui *Three State-Gate*.

Pembuatan mikroprogram untuk instruksi-instruksi sebagai berikut:

Load	LD	N →	[Accu] := N
Addition	A	N →	[Accu] := [Accu] + N
Subtraction	S	N →	[Accu] := [Accu] - N
Transfer	T	N →	N := [Accu]
Jumping	US	N →	Program Counter :=N
Jump Plus	BSP	N →	[Accu] =0 ? then PC =N Else PC=PC+1

Sementara itu, rangkaian yang detail kerjakan yaitu penguji rangkaian dan pengujian Mikroprogram dipersiapkan serta *control signal* dilakukan dengan tangan melalui saklar-saklar.

Rancangan *Controller* Sederhana



Gambar 2 Rancangan control sederhana

Keterangan PIN sinyal pengontrol:

$\overline{BZL}$ Operation Counter (loading)=Load Program Counter
$\overline{BZL}$ = 0 sinyal pada pin input dari program counter sama dengan sinyal pada pin output dari program counter dengan next clock yang aktif.

BZZ Program counter count
BZZ = 1 sinyal pada pin output = sinyal pada pin output+1 dengan next clock yang aktif.

BZS Program counter ke Memory
BZS = 1 Pin output dari program counter ke Address BUS melalui three-state driver.

BRL Load Instruction Register
$\overline{BRL}$ = 0 sinyal pada kaki input-instruction-register = sinyal pada kaki output-I-Register dengan next clock.

BRS Instruction Register ke Memori
BRS = 1 Output Instruction-Register ke Address-Bus melalui three-state-driver.

R/W Pengendali memori
R/W = 1 tulis
$\overline{R/W}$ = 0 baca isi memori

$\overline{SBZ}$ chipselect memori
$\overline{SBZ}$ = 0 output memori ke data-bus
SBZ = 1 output memori terpisah dengan data-bus (memori tak aktif)

BA Pengendali ALU

CN		$\overline{BA}$	CN	S0	S1	S2	S3
	ADD	0	1	1	0	0	1
	SUB	0	0	0	1	1	0
	F=B	1	*	0	1	0	1

AL Load accumulator
AL = 1 sinyal pada kaki input accumulator = sinyal pada kaki output accumulator dengan next clock.

AKS Accumulator ke memori
AKS = 1 sinyal pada kaki output accumulator = sinyal pada kaki input memori melalui three state driver.

TAS saklar ke memori
TAS = 1 input memori melalui 8 saklar.

TBZ saklar ke program counter
TBZ = 1 input ke program counter melalui 5 saklar

Mikroprogram untuk Suatu Percobaan

Tabel 1 Mikroprogram suatu percobaan

	Prog. Counter			Instr. Reg			Memory		ALU						ACCU			
OP code	**BZL**	**BZZ**	**BZS**	**BRL**	**BRS**	**BRZ**	**R/W**	**SBZ**	**BA**	**CN**	**S0**	**S1**	**S2**	**S3**	**AL**	**AKS**	**TAS**	**TBZ**
FETCH ****	1	0	1	0	0	0	0	0	*	*	*	*	*	*	0	*	0	0
ADD 1000	1	1	0	1	1	0	0	0	0	1	1	0	0	1	1	1	0	0
SUB 1000	1	1	0	1	1	0	0	0	0	0	0	1	1	0	1	1	0	0

LOAD 1010	1	1	0	1	1	0	0	0	1	*	0	1	0	1	1	1	0	0
Transfer																		
T1 1011	1	0	0	1	1	0	0	0	*	*	*	*	*	*	0	1	0	0
T2	1	0	0	1	1	0	1	0	*	*	*	*	*	*	0	1	0	0
T3	1	1	0	1	1	0	0	0	*	*	*	*	*	*	0	1	1	0
US	1	0	0	1	0	1	0	1	*	*	*	*	*	*	0	*	0	0
Jump 1100																		
BSP 1101																		
"+"	0	0	0	1	0	1	0	1	*	*	*	*	*	*	0	*	0	0
"-"	1	1	0	1	0	0	0	0	*	*	*	*	*	*	0	*	0	0

Keterangan *Program-Counter*:

BZL	BZZ			BZS	
0	0	Disable	1	ke address bus	
0	1	Stop	0	tidak ke address-bus	
1	0	Inhibit			
1	1	Count			

Instruction-Register

BRL

0 data-bus ke instruction-register

1 Instruction-register tak aktif

SIMPULAN

Bahwa dengan adanya rancangan *controller* sederhana ini akan menambah pengertian pada aplikasi dari komponen komponen yang digunakannnya, dan pengertian dari program mikro itu sendiri. Untuk menambah instruksi yang diperlukan perlu adanya program mikro tambahan. Rancangan *controller* yang mengandung banyak instruksi seperti yang ada sekarang ini, perlu tambahan komponen register, pengembangan alu dan lain lainnya.

DAFTAR PUSTAKA

Bartee, T. C. (1985). *Computer Digital fundamental*. New York: McGraw-Hill.

Hayes, J. P. (1979). *Computer Architecture and Organization*. Tokyo: McGraw-Hill International.

Stout, D. F. (1990). *Microprocessor Applications Handbook.* California: Santo Clara.

Mano, M. M. (1989). *Digital Logic and Computer Design*. New Delhi: Prentice Hall of India.