計測自動制御学会東北支部 第 265 回：2011 年 6 月 28 日
資料番号 265-1

組み込み制御用小規模プロセッサの開発

The Development of the Small Type Processor for Embedded Control Systems

○松平功介　一條健司　成田明子　吉岡良雄
○Kosuke Matsudaira, Kenji Ichijo, Akiko Narita, Yoshio Yoshioka

弘前大学大学院理工学研究科
Graduate School of Science and Technology, Hirosaki University

要旨：電子製品などは SOC(System On Chip)で制作され、それを制御するために組み込まれるプロセッサには、チップのハードウェア資源を有効に活用するために小規模なものが求められている。例として SONY の VME(Virtual Mobile Engine)技術における動的再構成デバイスを制御する小規模プロセッサが挙げられる。本報告では、組み込み制御用小規模プロセッサとして CPU1208 の開発を行い、その仕様と想定する制御、CPU コアの具体的な内部構成、プロセッサの具体的な実装について述べた。さらに、実装したプロセッサを活用するためのアセンブラ、動作させるためのアセンブリ言語によるモニタプログラムの紹介も行った。また、提案した小規模プロセッサ CPU1208 は CPLD MAX7000S EPM128SLC84-15 上に実装し、その使用マクロセル数は 89 と非常に小規模であることも述べた。
キーワード：CPU 設計(CPU Design)、CPLD(Complex Programmable Logic Device)、
アセンブラ(Assembler)、モニタプログラム(Monitor Program)、組み込みシステム(Embedded System)

連絡先：〒036-8561　青森県弘前市文京町 3 番地　弘前大学理工学部 1 号館 410 室
TEL/FAX: (0172)39-3669,　E-mail: h11gs517@stu.hirosaki-u.ac.jp

１．まえがき

近年の電子製品は SOC(System On Chip)で制作され、チップ内のハードウェア資源を有効活用するために、電子製品を制御するために組み込まれる CPU には小規模なものが求められている。例として SONY の VME(Virtual Mobile Engine)技術における動的再構成デバイスを制御する小規模プロセッサが挙げられる。そこで本報告では、組み込み制御用小規模プロセッサとして CPU1208(アドレスバス 12-bit・8-bitCPU)を開発し、その仕様と実現方法について述べる。また、CPU1208 を活用するためのソフトウェアについても述べる。ここで、CPU1208 は ALTERA 社の開発ツール Quartus Ⅱ 9.1sp2 Web Edition を用いて、ハードウェア記述言語 VHDL(Very high speed integrated circuit Hardware Description Language)によって記述する。また実装は ALTERA 社の CPLD MAX7000S EPM128SLC84-15 に行う。

２．CPU1208 の仕様

本報告で示す小規模プロセッサ CPU1208 は単純な制御を想定して構成している。例えば、通信制御やセンサの信号に対するモータ等の制御であ

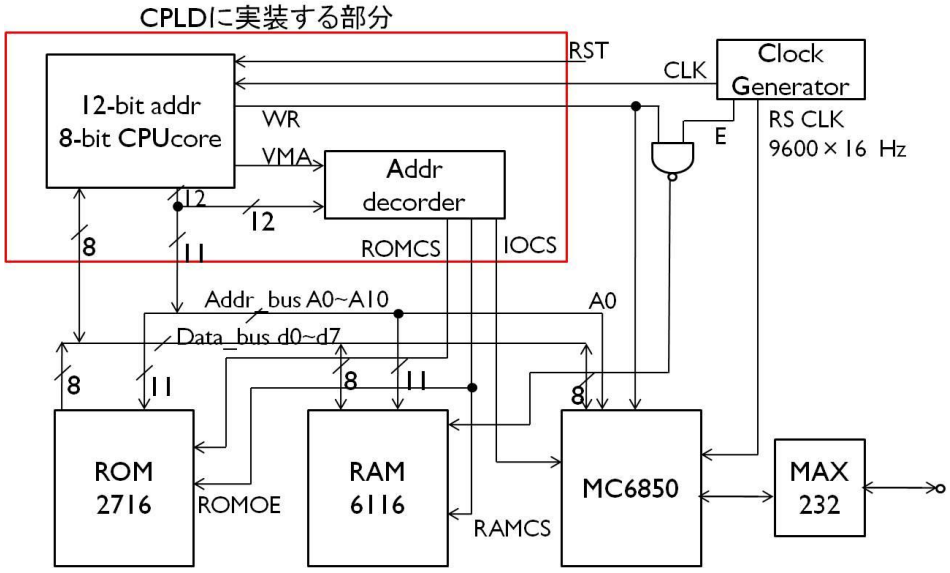

図１．CPU1208 を用いたコンピュータシステム

る。ここでは開発した小規模プロセッサの仕様について述べる。

CPU1208 を用いたコンピュータシステムを図１に示す。また、この CPU1208 の仕様は以下のようである。

[CPU1208 の仕様]

①アドレス空間は 12-bit である。

・$000~$7F7　　：ROM 領域

・$7F8~$7FF　　：I/O 領域

・$800~$FFF　　：RAM 領域

②汎用レジスタは 8-bit レジスタ１つである。

③フラグは演算結果が 0 の時に１になる Z フラグのみである。

④機械語命令は 16 種類あり、命令セットを表１に示す。これらの命令は先に述べた制御をするために必要な最低限の命令+α である。命令長は 16-bit と 24-bit の２種類ある。まず、ロード命令は３種類あり、即値型ロード(lda)、拡張型ロード(ldax)、間接型ロード(ldai)である。続いてストア命令は２種類で、拡張型ストア(stax)、間接型ストア(stai)である。演算命令は数値加算、論理積、論理和、排他的論理和の 4 種類でそれぞれに即値型(adda,anda,ora,eora)と拡張型(addax,andax,orax,eorax)がある。これらの演算命令の内、拡張型の andax, orax が前述した+α で、本来なら必要ではないが実装していても CPU のハードウェア量などに与える影響はないので、現在は実装している。これらは今後、別の命令と置き換え可能である。また、分岐命令として無条件分岐(jmp)、ビットテスト条件分岐(bitajze)、即値データ比較条件分岐(equajze)がある。

表１．実装している機械語命令

機械語			操作			ニーモニック
00	nn		$nn	→	A	lda
1a	dr		adr	→	PC	jmp
2a	dr		(adr)	→	A	ldax
3a	dr		A	→	(adr)	stax
40	nn		A + $nn	→	A	adda
50	nn		A and $nn	→	A	anda
60	nn		A or $nn	→	A	ora
70	nn		A eor $nn	→	A	eora
8a	dr		A + (adr)	→	A	addax
9a	dr		A and (adr)	→	A	andax
Aa	dr		A or (adr)	→	A	orax
Ba	dr		A eor (adr)	→	A	eorax
Ca	d1	d2	[(adr)]	→	A	ldai
Da	dr	nn	adr	→	PC when(A and $nn =0)	bitajze
Ea	d1	d2	A	→	[(adr)]	stai
Fa	dr	nn	adr	→	PC when(A eor $nn =0)	equajze

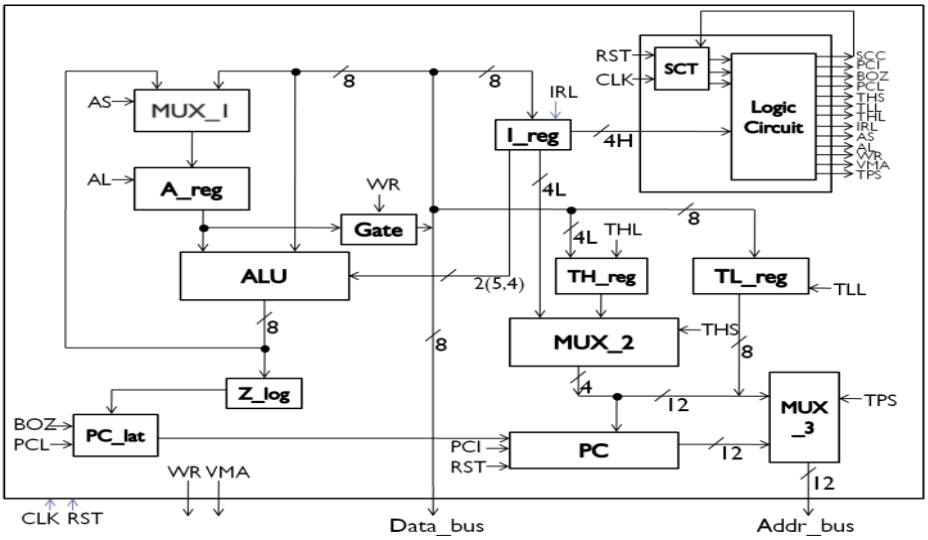

図２．CPU コアの構成図

３．CPU コアの内部構成

２章の４つの仕様に基づいてCPUコアの設計を行った。その構成図が図２である。

仕様②で述べた8-bitの汎用レジスタはA_regである。A_regに入力される信号にはALUの出力値と内部バスからの値があり、MUX_1によってどちらかの値のみが選択される。命令レジスタ I_regは8-bitレジスタでメモリから取り出したデータを格納する。その時、上位4-bitをオペレーションコード、下位4-bitをオペランドアドレス12-bitの内の上位4-bitとする。TH_regは4-bitレジスタで、間接型の命令を実行するときに実効アドレスの上位4-bitを格納する。TL_regは8-bitレジスタで、オペランドアドレス12-bitの内の下位8-bit、また、間接型命令の時は実効アドレスの下位8-bitが格納される。PCは12-bitのプログラムカウンタである。ALUは8-bit演算機でA_regの値と内部バスの値の演算をする。演算の種類は表２で示している４種類で加算、論理積、論理和、排他的論理和である。Z_logは仕様３で述べたZフラグを生成する論理回路である。PC_latは分岐命令の際にPCにラッチ信号を送る論理回路である。制御部はワイヤードロジック方式であり、SCT(Step CounTer)とI_reg上位4-bitの値を入力信号として制御信号を生成する論理回路から成っている。

表２．ALU機能

	演算			IR(5)	IR(4)
A ＝	A	+	ID	0	0
	A	and	ID	0	1
	A	or	ID	1	0
	A	eor	ID	1	1

続いてCPUの状態遷移図を図３に、各状態での操作とそれに対応した命令・動作を表３に示す。

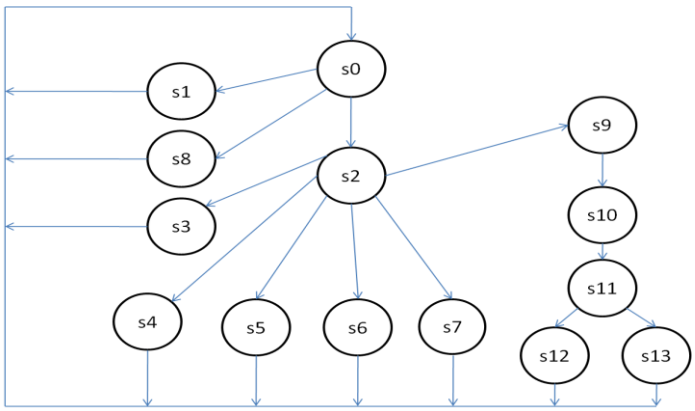

図３．状態遷移図

表３．各状態での操作

状態	操作	命令・動作
s0	M(PC)→IR, PC+	IR_latch
s1	M(PC)→A, PC+	lda
s2	M(PC)→TRL, PC+	Addr_latch
s3	IRL&TRL→PC	jmp
s4	M(IRL&TRL)→A	ldax
s5	A→M(IRL&TRL)	stax
s6	A calc M(IRL&TRL)→A	calcx
s7	A calc M(IRL&TRL), IRL&TRL→PC	jze
s8	A calc M(IRL&TRL)→A, PC+	calc
s9	M(IRL&TRL)→TRH	ldai,stai_1
s10	M(PC)→TRL, PC+	ldai,stai_2
s11	M(IRL&TRL)→TRL	ldai,stai_3
s12	M(TR)→A	ldai_4
s13	A→M(TR)	stai_4

まず CPU は s0 でメモリからデータをロードし I_reg にラッチする。そして I_reg の値から次の状態を決定し遷移する。次に各状態での制御信号を表 4 に示す。

表 4 の制御信号は図 2 の制御信号と対応している。TPS(TemP register Select)はアドレスバスに信号を出力する際に、0 の時 PC の値を、1 の時(I_reg or TH_reg)&TL_reg の値を選択する。

表 4．各状態での制御信号

状態	TPS	VMA	WR	AL	AS	IRL	THL	TLL	THS	PCL	BOZ	PCI	SCC
s0	0	1	0	0	0	1	0	0	0	0	0	1	0
s1	0	1	0	1	0	0	0	0	0	0	0	1	1
s2	0	1	0	0	0	0	0	1	0	0	0	1	0
s8	0	1	0	1	1	0	0	0	0	0	0	1	1
s3	0	0	0	0	0	0	0	0	0	1	0	0	1
s4	1	1	0	1	0	0	0	0	0	0	0	0	1
s5	1	1	1	0	0	0	0	0	0	0	0	0	1
s6	1	1	0	1	1	0	0	0	0	0	0	0	1
s7	0	1	0	0	0	0	0	0	0	0	1	1	1
s9	1	1	0	0	0	0	1	0	0	0	0	0	0
s10	0	1	0	0	0	0	0	1	0	0	0	1	0
s11	1	1	0	0	0	0	0	1	0	0	0	0	0
s12	1	1	0	1	0	0	0	0	1	0	0	0	1
s13	1	1	1	0	0	0	0	0	1	0	0	0	1

VMA(Valid Memory Address)は 1 の時にアドレスバス上の信号が有効になる。WR(WRite)は 1 の時に A レジスタの内容を内部バスに出力する。AL(A register Latch)が 1 の時、MUX_1 の出力信号を取りこむ。AS(A register Select)は 0 の時に内部バス信号を、1 の時に ALU の出力信号を選択する。IRL(Instruction Register Latch)は 1 の時に内部バス信号を取りこむ。THL(Temp register High Latch)は 1 の時に内部バス信号の下位 4-bit を TH_reg に取り込む。TLL(Temp register Low Latch)は 1 の時に内部バス信号を取り込む。THS(Temp register High Select)は 0 の時 I_reg の下位 4-bit を、1 の時 TH_reg を選択する。PCL(Program Counter Latch)は 1 の時、PC は I_reg の下位 4-bit と TL_reg を取り込む。BOZ(Branch On Zeroflag)は、この信号が 1 の時に Z_log の出力信号が 1 であれば、PC は I_reg 下位 4-bit と TL_reg を取り込む。PCI(Program Counter Increment)は 1 の時に PC の値を 1 増加させる信号である。SCC(Step Counter Clear)は 1 の時に SCT の値をクリアする信号である。

4．CPU1208 の具体的な実装方法

ここでは図 1 で示したコンピュータシステムを実現したものを示す。それが図 4 である。

前述したように CPU コア(アドレスデコーダ含む)は VHDL で記述し、CPLD MAX 7000S EPM128SLC84-15 に実装している。その使用マクロセル数は 89 である。

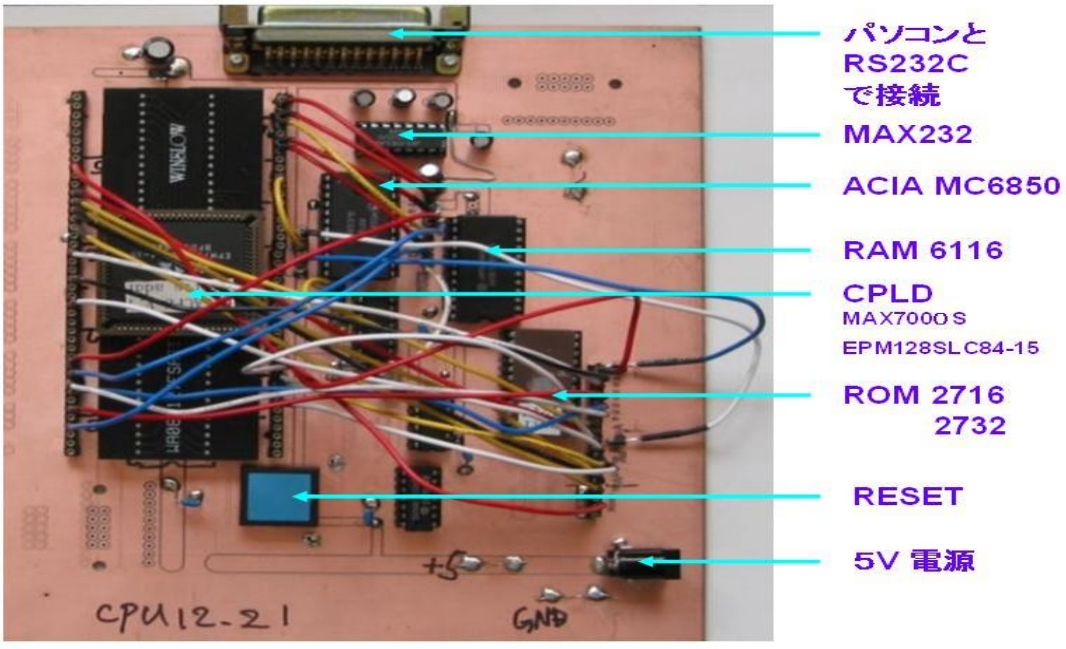

図 4．CPU1208 の具体的実装

５．ソフトウェア

この章では開発した CPU1208 を活用するためのソフトウェアを紹介する。

5.1. アセンブラ

プログラムを記述する際に機械語命令コードでは非常に記述しづらい。そこで、本システム用のアセンブラを開発した。ここではアセンブリプログラムから機械語プログラムに変換するアセンブラについて、その中で定義しているマクロ命令と擬似命令を紹介する。マクロ命令が表５、擬似命令が表６である。また、次の 5.2 節ではモニタプログラムのアセンブリ記述も図５で示している。

表５．マクロ命令

命令記述	アセンブリ言語記述		機械語命令
call adr2,adr1	lda	#$1m	00 1m
	stax	adr1	3a dr1
	lda	#$nn	00 nn
	stax	adr1+1	3a dr1+1
	jmp	adr2	1a dr2
ret sp	jmp	adr	1a dr
move8 #$nn,addr	lda	#$nn	00 nn
	stax	adr	3a dr
move8 adr1,adr2	ldax	adr1	2a dr1
	stax	adr2	3a dr2
move16 #$mmnn,adr	lda	#$mm	00 mm
	stax	adr	3a dr
	lda	#$nn	00 nn
	stax	adr+1	3a dr+1
move16 adr1,adr2	ldax	adr1	2a dr1
	stax	adr2	3a dr2
	ldax	adr1+1	2a dr1+1
	stax	adr2+1	3a dr2+1
inc1 adr	ldax	adr	2a dr
	adda	#$01	40 01
	stax	adr	3a dr
dec1 adr	ldax	adr	2a dr
	adda	#$ff	40 ff
	stax	adr	3a dr

表５はマクロ命令で、命令の種類としてサブルーチンコール(call)と戻り命令(ret)、8-bit データ転送(move8)、16-bit データ転送(move16)、1 加算(inc1)、1 減算(dec1)がある。なお、表５の call 命令での戻り番地はアセンブラで求めている。

表６は擬似命令を示していて、メモリのどの番地からプログラムを格納するかを指定する org 命令と、等価命令 equ、定数の記憶領域を確保する dc、dw がある。

表６．擬似命令

命令記述	機能
org adr	プログラム先頭アドレス指定
label equ adr	等価命令(label = adr)
dc nn	1-byte定数の記憶領域確保
dc 'abc---'	文字列定数の記憶領域確保
dw adr	2-byte定数の記憶領域確保

5.2. モニタプログラム

続いて CPU1208 を動作させるためのモニタプログラムを紹介する。そのアセンブリプログラムの一部を抜粋したのが図５である。

```
1  000                    ( 1) ************************
2  000                    ( 2) * Monitor for 12 bitts CPU *
3  000                    ( 3) * made by Y.Yoshioka      *
4  000                    ( 4) * ver 1.08mx (2010.11.26) *
5  000                    ( 5) ************************
6  000                    ( 6)  org $0000
7  000                    ( 7) acia0 equ $7fe
8  000                    ( 7) lrdbf equ $ec0
 .
 .
70  060                   ( 38) lprt0
71  060  3F C1            ( 39)  stax Xreg0+1
72  062  00 10 3F EC      ( 39)  call lprx1,sp_put
     066  00 6C 3F ED
     06A  10 BE
73  06C  2F F3 40 01 3F F3 ( 39)  inc1 pr_addr+1
 .
 .
107  0BE                  ( 57) lprx1
108  0BE  27 FE           ( 58)  ldax acia0
 .
 .
731  785                  (380)  end
```

図５．モニタプログラムのアセンブリ記述

図６はモニタプログラムの操作を表示した図である。モニタプログラムには表 7 に示す 3 つのコマンドを実装している。

表７．モニタプログラムコマンド

コマンド	動作
d	メモリダンプ
s	メモリ内容変更・プログラム格納
g	プログラム実行

図６では s コマンドで$800 番地から機械語命令を格納し、d コマンドでメモリダンプを実行している。そして g コマンドで$800 番地から格納したプログラムを実行している。ここで実行しているプログラムはハノイの塔のプログラムである。

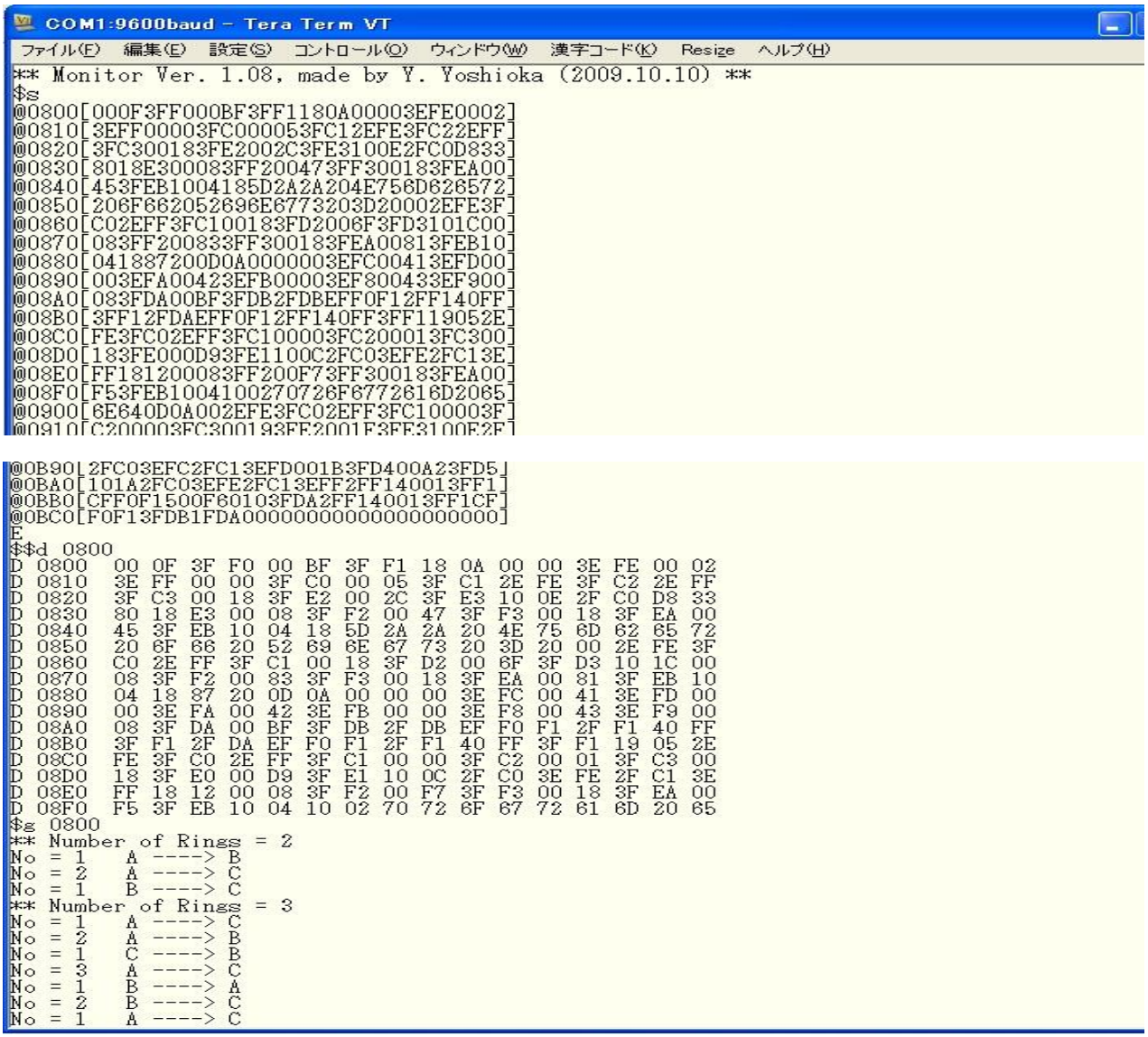

図６．モニタプログラムの操作

６．まとめ

本報告では、開発した小規模プロセッサ CPU1208 のコンピュータシステムと想定する制御を述べ、仕様を示し、CPU コアの具体的な内部構成とコンピュータシステムの具体的な実装を示した。また、CPU1208 を活用するためのソフトウェアとして、アセンブラで定義しているマクロ命令と擬似命令、CPU1208 をコマンドで動作させるモニタプログラムを紹介した。

また、この CPU は小規模(マクロセル数：89)であり、通信制御などの単純な制御をするために必要な命令は実装されているので、組み込み制御用プロセッサとして有用である。

今後は、対象を定めての実際の制御や、前述した仕様に基づいての、種々の構成の CPU の設計や、VHDL 記述の仕方によるさらなる小規模化を行う。

参考文献

[1]吉岡良雄「手作り CPU(ハードウェア記述言語 VHDL による)」、弘前大学出版会、(2006)

[2]Virtual Mobile Engine(VME)「”変身する LSI”が超低消費電力・多機能化を実現」http://www.sony.co.jp/Products/SC-HP/cx_pal/vol66/pdf/sideview.pdf

[3]吉岡良雄、一條健司「ハードウェア設計・演習(基礎からプロセッサ設計まで)」、弘前大学出版会、(2010)

[4]Altera Corporation「MAX 7000 Programmable Logic Device Family Data Sheet」