

2. 転送命令を学ぼう

2004年8月に本講義ノートをWebにアップして以来、とても多くの方の訪問を受けてきました。内容が一部古くなっていたので、2012年5月時点の情報に書き改めました。主な変更点は以下の通りです。

第0章に本講座の準備のための章を設け、以下の更新をしました。

1. プログラム開発環境(MPLAB IDE)をv8.84に更新しました。
2012年5月時点での最新バージョンはMPLAB X IDE v1.10ですが、アセンブラの勉強のためには、旧バージョンのv8.84が良さそうです。（筆者がXを使いこなしていないこともあります。）
2. ブレッドボード上に回路を製作し、その詳細を記しました。
ハンダ付けをほとんど必要とせずにマイコン回路を製作できます。
3. 部品を全てネットで購入して、その仕様と入手先を記しました。
入手が容易な部品ばかりでマイコン回路を製作しました。
4. In-Circuit Debugger/ProgrammerにPICkit3を用いました。
ブレッドボード上のマイコンへのプログラミングにPICkit3を使用した例を記しました。

; Load of Literal to Working Register

```
INCLUDE"p16F84A.inc"  
list p=16F84A
```

```
__CONFIG _HS_OSC & _WDT_OFF & _PWRTE_OFF & _CP_OFF
```

```
ORG 0  
GOTO START  
ORG 4
```

START

```
MOVLW B'10011010' ; Binary number
```

```
MOVLW 0xAB ; Hexadecimal number
```

```
MOVLW D'255' ; Decimal number
```

```
GOTO START
```

```
END
```

まず、このソースファイルを打ち込んで下さい。
MPLAB IDE v8.84の立ち上げ方、Project Fileの作り方、アセンブラのソースファイルの作り方、ビルドの仕方、シミュレーションの仕方は[第0章](#)を参照してください。 **注意** CONFIGの前のアンダーバーは2つあります。

これはWレジスタにLiteral (リテラル、文字) を転送させるプログラムです。最初に2進数の10011010を転送し、次に16進数のABを転送し、最後に10進数の255を転送します。

MOVLWは**Move Literal to Working register**の略です。

シミュレーションの仕方

プログラムを打ち込み終わったら、**Debugger** → **Select Tool** → **MPLAB SIM** → **Project** → **Make**と選択すると、プログラムに文法上の間違いがなければ、**BUILD SUCCEEDED**というメッセージが出ます。次に**View** → **Special Function Registers**を選択して下さい。下図のようにレジスタ群の内容を示すウィンドウが開かれます。図示の位置にカーソルを持ってきて右クリックすると、レジスタ内容の表示型をHex(16進数)、Binary(2進数)、Decimal(10進数)、Char(文字型)の中から選択できます。

The screenshot shows the MPLAB IDE interface. The main window displays the assembly code for a PIC16F84A. The **Special Function Registers** window is open, showing a list of registers. The **Output** window is also open, displaying the build output, including the message **BUILD SUCCEEDED**. A callout box with a blue border and text points to the **WREG** register in the SFR list.

Address	SFR Name	Hex
00	WREG	0x00
01	INDF	--
02	TMR0	0x00
03	PCL	0x00
04	STATUS	0x18
05	FSR	0x00
06	PORTA	0x00
07	PORTB	0x00
08	EEDATA	0x00
09	EEADR	0x00
0A	PCLATH	0x00
0B	INTCON	0x00
81	OPTION_REG	0xFF
85	TRISA	0x1F
86	TRISB	0xFF
88	EECON1	0x00
89	EECON2	0x00

DecimalとBinary
を表示

Address	SFR Name	Hex	Decimal	Binary
00	WREG	0x00	0	00000000
01	INDF	--	-	-
02	TMR0	0x00	0	00000000
03	PCL	0x00	0	00000000
04	STATUS	0x18	24	00011000
05	FSR	0x00	0	00000000
06	PORTA	0x00	0	00000000
07	PORTB	0x00	0	00000000
08	EEDATA	0x00	0	00000000
09	EEADR	0x00	0	00000000
0A	PCLATH	0x00	0	00000000
0B	INTCON	0x00	0	00000000
81	OPTION_REG	0xFF	255	11111111
85	TRISA	0x1F	31	00011111
86	TRISB	0xFF	255	11111111
88	EECON1	0x00	0	00000000
89	EECON2	0x00	0	00000000

Wレジスタ
内の数字が
表示されて
います。

Output

```

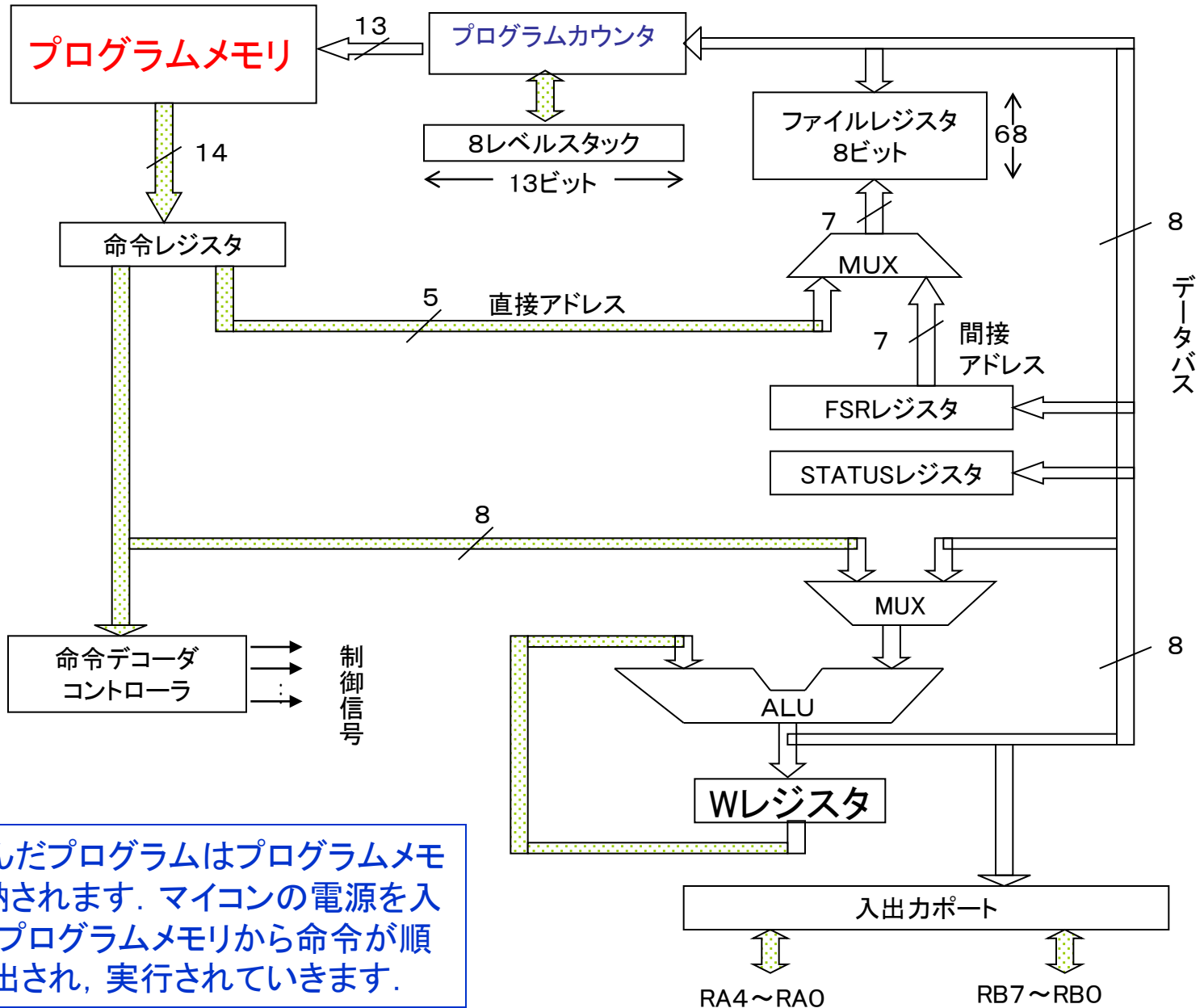
Build Version Control Find in
Loaded C:\Program Files (x
-----
Debug build of project
Language tool versions
Preprocessor symbol \
Wed May 09 11:32:07 20
-----
BUILD SUCCEEDED

```

MPLAB SIM PIC16F84A pc:0 W:0 z dc c 20 MHz bank 0

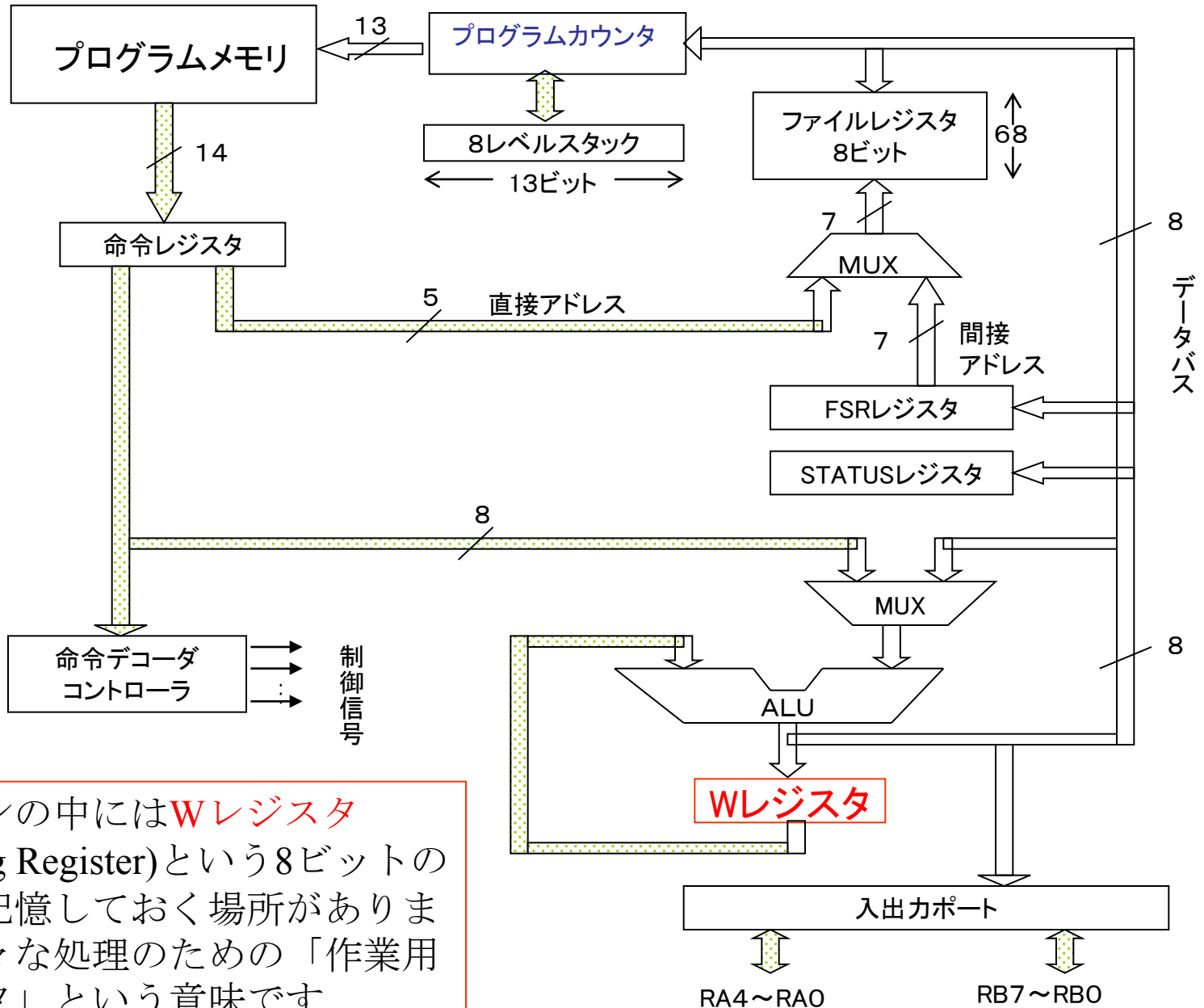
ファンクションキーのF7を押して、プログラムを一行ずつ実行して、WREG (Wレジスタ) 内の数字がどう書き換えられていくかを確認してください。

PIC16F84AのData SheetのBLOCK DIAGRAMの抜粋です。



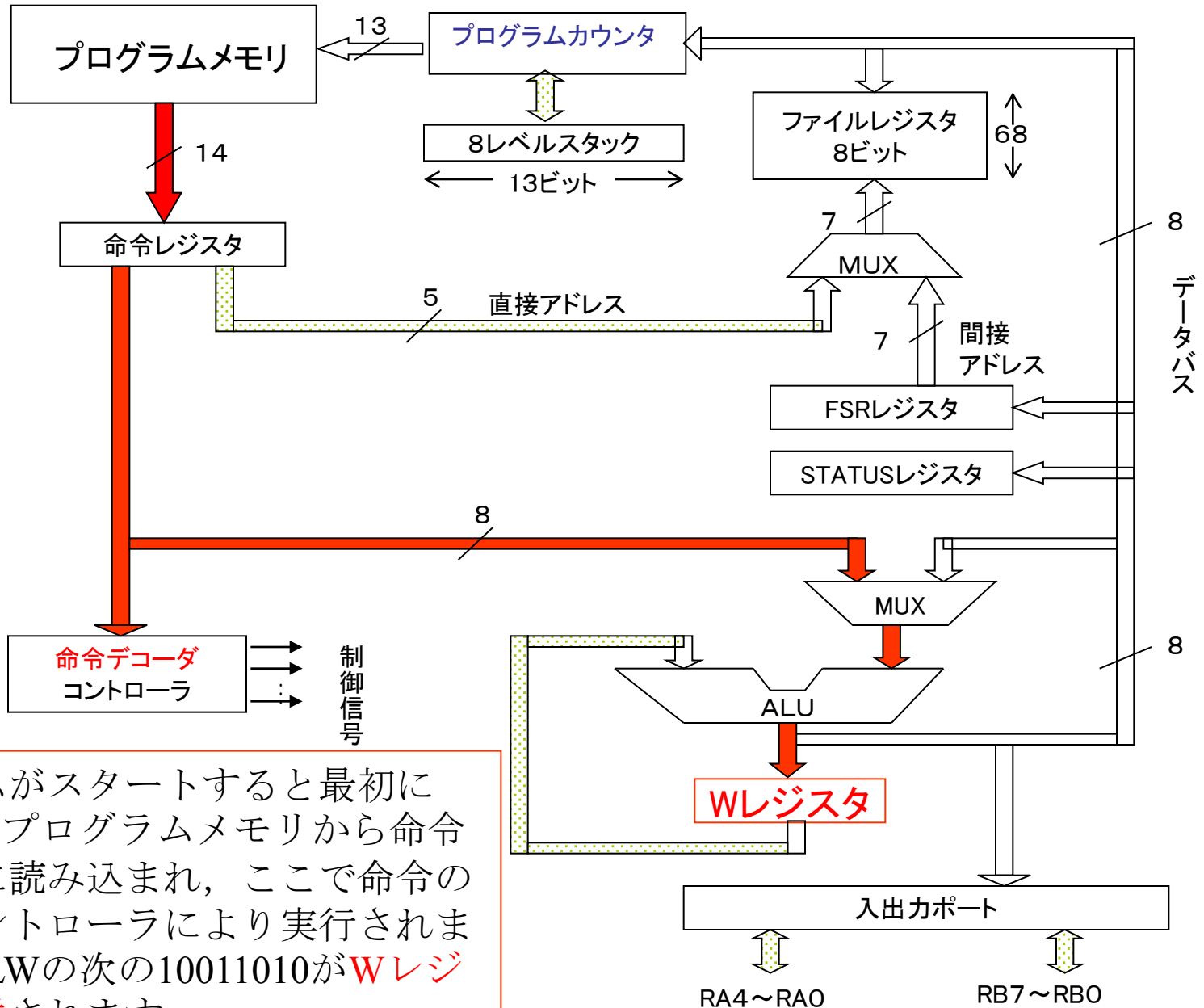
打ち込んだプログラムはプログラムメモリに格納されます。マイコンの電源を入れると、プログラムメモリから命令が順次読み出され、実行されていきます。

PIC16F84AのData SheetのBLOCK DIAGRAMの抜粋です。



マイコンの中には**Wレジスタ** (Working Register) という8ビットの数字を記憶しておく場所があります。様々な処理のための「作業用レジスタ」という意味です。

MOVLW B'10011010' の実行



プログラムがスタートすると最初に MOVLW がプログラムメモリから命令デコーダに読み込まれ、ここで命令の解釈、コントローラにより実行されます。MOVLW の次の 10011010 が **Wレジスタ** に転送されます。

; Load of Literal to Working Register

```
INCLUDE "p16F84A.inc"  
list p=16F84A
```

Configuration設定です。マイコンのシステムクロックを設定します。

```
__CONFIG _HS_OSC & _WDT_OFF & _PWRTE_OFF & _CP_OFF
```

```
ORG    0  
GOTO   START  
ORG    4  
  
START  
  
MOVLW   B'10011010'  
  
MOVLW   0xAB  
  
MOVLW   D'255'  
  
GOTO    START  
  
END
```

MicrochipのWebページからダウンロードできるPIC16F84AのData Sheetの21ページにConfigurationの説明があります。

OSC: Oscillator

RC: Resistor/Capacitor

内蔵の発振回路, 4MHz程度まで
発信周波数は不安定

HS: High Speed Crystal/Resonator

3.5MHz 以上で推奨

本回路では10MHzのセラミック発振子を外付けしてあるので, HS設定とする。

XT: Crystal/Resonator

4 MHz 以下

LP: Low Power Crystal

低消費電力, 200kHz 以下

WDT: Watch Dog Timer

PWRTE: Power up Timer Enable

CP: Code Protection

本資料の使い方の範囲内では全てオフで差し支えない。

; Load of Literal to Working Register

```
INCLUDE "p16F84A.inc"  
list p=16F84A
```

```
__CONFIG _HS_OSC & _WDT_OFF & _PWRTE_OFF & _CP_OFF
```

```
ORG 0
```

```
GOTO START
```

```
ORG 4
```

START

```
MOVLW B'10011010'
```

```
MOVLW 0xAB
```

```
MOVLW D'255'
```

```
GOTO START
```

```
END
```

現実のマイコンでは電源を入れる
ところから始まります。

STARTへジャンプする。

; Load of Literal to Working Register

```
INCLUDE "p16F84A.inc"
```

```
list p=16F84A
```

```
__CONFIG _HS_OSC & _WDT_OFF & _PWRTE_OFF & _CP_OFF
```

```
ORG 0
```

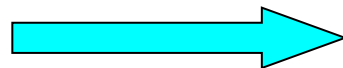
```
GOTO START
```

```
ORG 4
```

wレジスタに100110101という数を転送せよという命令です。

START

```
MOVLW B'10011010'
```



Wレジスタ

10011010

```
MOVLW 0xAB
```

```
MOVLW D'255'
```

```
GOTO START
```

```
END
```

B'10011010' 2進数は
= D'154' 10進数
= H'9A' 16進数です。
表現が違うだけでどれも同じ
数字です。

; Load of Literal to Working Register

```
INCLUDE "p16F84A.inc"
```

```
list p=16F84A
```

```
__CONFIG _HS_OSC & _WDT_OFF & _PWRTE_OFF & _CP_OFF
```

```
ORG 0
```

```
GOTO START
```

```
ORG 4
```

START

```
MOVLW B'10011010'
```

```
MOVLW 0xAB
```

```
MOVLW D'255'
```

```
GOTO START
```

```
END
```

wレジスタにABという数を転送せよという命令です。

Wレジスタ

10101011

B'10101011' 2進表現
= D'171' 10進表現
= H'AB' 16進表現

; Load of Literal to Working Register

```
INCLUDE "p16F84A.inc"
```

```
list p=16F84A
```

```
__CONFIG _HS_OSC & _WDT_OFF & _PWRTE_OFF & _CP_OFF
```

```
ORG 0
```

```
GOTO START
```

```
ORG 4
```

START

```
MOVLW B'10011010'
```

```
MOVLW 0xAB
```

```
MOVLW D'255'
```

```
GOTO START
```

```
END
```

wレジスタに255という数を転送せよという命令です。

Wレジスタ

11111111

B'11111111' 2進表現
= D'255' 10進表現
= H'FF' 16進表現

; Load of Literal to Working Register

```
INCLUDE "p16F84A.inc"  
list p=16F84A
```

```
__CONFIG _HS_OSC & _WDT_OFF & _PWRTE_OFF & _CP_OFF
```

```
ORG 0  
GOTO START  
ORG 4
```

START

```
MOVLW B'10011010'
```

```
MOVLW 0xAB
```

```
MOVLW D'255'
```

```
GOTO START
```

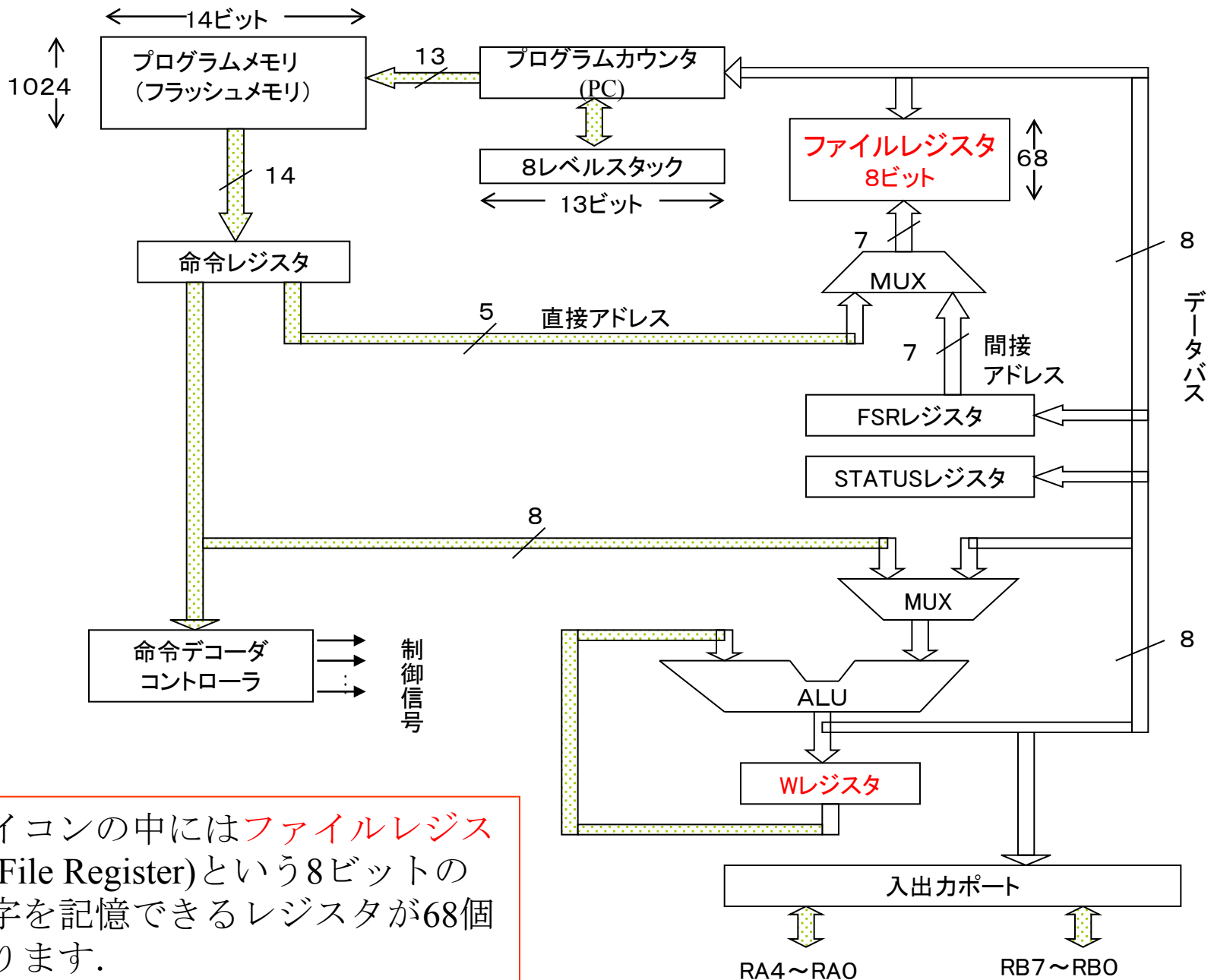
```
END
```

Wレジスタ

11111111

STARTへジャンプする.

次に、ワーキング(W)レジスタと
ファイルレジスタ間でのデータの
やりとりを体験してください。



マイコンの中には**ファイルレジスタ (File Register)**という8ビットの数字を記憶できるレジスタが68個あります.

PIC16F84Aの構成

0C~4Fの番地を持つ68個の8ビットレジスタがある.

8ビットの数字を68個までしまっておける「引き出し」のようなもの.

データの「一時保管所」のような意味です.

0x0C



0x4F

ファイルレジスタ

Wレジスタはたった1個の8ビットレジスタからなる.



wレジスタ

; Data from Working Register to File Register and vice versa

```
INCLUDE"p16F84A.inc"
list p=16F84A

__CONFIG _HS_OSC & _WDT_OFF & _PWRTE_OFF & _CP_OFF

MEM1 EQU 0x0C ;MEM1 at 0C
MEM2 EQU 0x0C+1 ;MEM2 at 0D

ORG 0
GOTO START
ORG 4

START

MOVLW 0x9A ; '9A' to Working Register

MOVWF MEM1 ; Move '9A' to MEM1

MOVLW 0x01 ; '01' to Working Register

MOVWF MEM2 ; Move '01' to MEM2

MOVF MEM1,0 ; Load '9A' from MEM1 to W

MOVF MEM2,0 ; Load '01' from MEM2 to W

GOTO START

END
```

このプログラムを
打ち込んで下さい。

シミュレータでプロ
グラムをステップ
実行しながら、W,
MEM1, MEM2レ
ジスタの中の数字
がどう変わるか確
認して下さい。

MEM1, MEM2の中の数字は, View → File Registersにより見ることができます。

The screenshot displays the MPLAB IDE interface with the following components:

- Assembly Code Window:** Shows assembly instructions for PIC16F84A. It defines MEM1 at address 0C and MEM2 at address 0D. The code includes instructions like MOVWF MEM1, MOVWF MEM2, MOVF MEM1,0, and MOVF MEM2,0.
- Special Function Registers Window:** A table listing SFRs with their addresses, names, hex, decimal, and binary values.
- File Registers Window:** A table listing file registers with their addresses, hex, decimal, binary, and symbol names. MEM1 and MEM2 are highlighted with red lines.

Address	SFR Name	Hex	Decimal	Binary
00	WREG	0x00	0	00000000
01	INDF	--	-	-
02	TMR0	0x00	0	00000000
03	PCL	0x00	0	00000000
04	STATUS	0x18	24	00011000
05	FSR	0x00	0	00000000
06	PORTA	0x00	0	00000000

Address	Hex	Decimal	Binary	Symbol Name
00	--	-	-----	INDF
01	0x00	0	00000000	TMR0
02	0x00	0	00000000	PCL
03	0x18	24	00011000	STATUS
04	0x00	0	00000000	FSR
05	0x00	0	00000000	PORTA
06	0x00	0	00000000	PORTB
07	--	-	-----	GIE
08	0x00	0	00000000	EEDATA
09	0x00	0	00000000	EEADR
0A	0x00	0	00000000	PCLATH
0B	0x00	0	00000000	INTCON
0C	0x00	0	00000000	MEM1
0D	0x00	0	00000000	MEM2
0E	0x00	0	00000000	

0C番地にMEM1
0D番地にMEM2

; Data from Working Register to File Register and vice versa

```
INCLUDE"p16F84A.inc"  
list p=16F84A
```

```
__CONFIG _HS_OSC & _WDT_OFF & _PWRTE_OFF & _CP_OFF
```

```
MEM1 EQU 0x0C ;MEM1 at 0C  
MEM2 EQU 0x0C+1 ;MEM2 at 0D  
  
ORG 0  
GOTO START  
ORG 4  
  
START  
MOVLW 0x9A ; '9A' to Working Register  
  
MOVWF MEM1 ; Move '9A' to MEM1  
  
MOVLW 0x01 ; '01' to Working Register  
  
MOVWF MEM2 ; Move '01' to MEM2  
  
MOVF MEM1,0 ; Load '9A' from MEM1 to W  
  
MOVF MEM2,0 ; Load '01' from MEM2 to W  
  
GOTO START  
  
END
```

0C番地のファイルレジスタにMEM1という名前をつけ、0C+1 (= 0D)番地のファイルレジスタにMEM2という名前を付けます。

; Data from Working Register to File Register and vice versa

```
INCLUDE"p16F84A.inc"
list p=16F84A

__CONFIG __HS_OSC & __WDT_OFF & __PWRTE_OFF & __CP_OFF

MEM1 EQU 0x0C ;MEM1 at 0C
MEM2 EQU 0x0C+1 ;MEM2 at 0D

ORG 0
GOTO START
ORG 4

START
MOVLW 0x9A ;'9A' to Working Register
MOVWF MEM1 ; Move '9A' to MEM1
MOVLW 0x01 ;'01' to Working Register
MOVWF MEM2 ; Move '01' to MEM2
MOVF MEM1,0 ; Load '9A' from MEM1 to W
MOVF MEM2,0 ; Load '01' from MEM2 to W

GOTO START

END
```

電源を入れるとORG 0の次から始まります.

STARTへジャンプする.

; Data from Working Register to File Register and vice versa

```
INCLUDE"p16F84A.inc"  
list p=16F84A
```

```
__CONFIG _HS_OSC & _WDT_OFF & _PWRTE_OFF & _CP_OFF
```

```
MEM1 EQU 0x0C  
MEM2 EQU 0x0C+1
```

```
ORG 0  
GOTO START  
ORG 4
```

START

```
MOVLW 0x9A
```

```
MOVWF MEM1
```

```
MOVLW 0x01
```

```
MOVWF MEM2
```

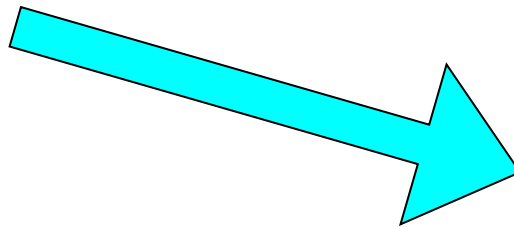
```
MOVF MEM1,0
```

```
MOVF MEM2,0
```

```
GOTO START
```

```
END
```

Wレジスタに
100110101という数を
転送せよという命令



Wレジスタ

10011010

; Data from Working Register to File Register and vice versa

```
INCLUDE "p16F84A.inc"  
list p=16F84A
```

```
__CONFIG __HS_OSC & __WDT_OFF & MEM1
```

```
MEM1 EQU 0x0C  
MEM2 EQU 0x0C+1
```

```
ORG 0  
GOTO START  
ORG 4
```

START

```
MOVLW 0x9A
```

```
MOVWF MEM1
```

```
MOVLW 0x01
```

```
MOVWF MEM2
```

```
MOVF MEM1,0
```

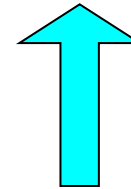
```
MOVF MEM2,0
```

```
GOTO START
```

```
END
```

ファイルレジスタ

10011010



Wレジスタ

10011010

wレジスタの内容をファイルレジスタのMEM1に転送せよという命令です。
MOVE W to F の略です。

; Data from Working Register to File Register and vice versa

```
INCLUDE "p16F84A.inc"  
list p=16F84A
```

```
    __CONFIG _HS_OSC & _WDT_OFF & MEM1  
MEM1 EQU 0x0C  
MEM2 EQU 0x0C+1  
  
ORG 0  
GOTO START  
ORG 4  
  
START  
    MOVLW 0x9A  
    MOVWF MEM1  
    MOVLW 0x01  
    MOVWF MEM2  
    MOVF MEM1,0  
    MOVF MEM2,0  
  
    GOTO START  
  
END
```

Wレジスタに
00000001という
数を転送せよと
いう命令

ファイルレジスタ

10011010

Wレジスタ

00000001

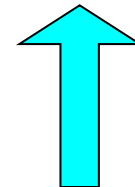
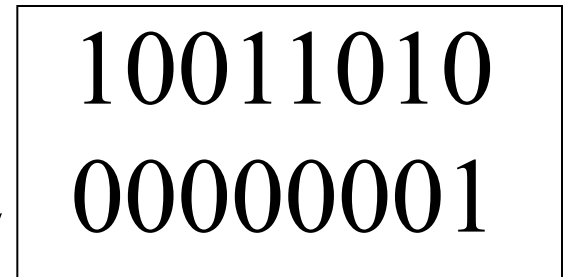
; Data from Working Register to File Register and vice versa

```
INCLUDE "p16F84A.inc"  
list p=16F84A
```

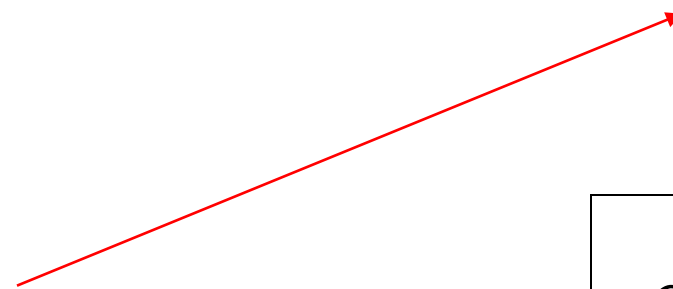
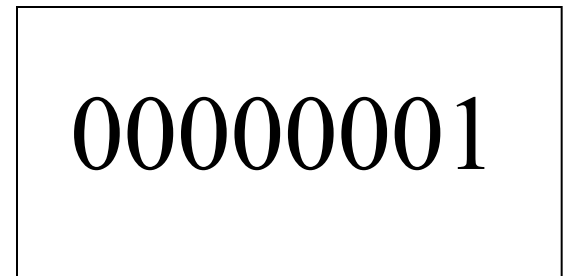
```
__CONFIG _HS_OSC & _WDT_OFF &  
MEM1 EQU 0x0C  
MEM2 EQU 0x0C+1  
ORG 0  
GOTO START  
ORG 4  
START  
MOVLW 0x9A  
MOVWF MEM1  
MOVLW 0x01  
MOVWF MEM2  
MOVF MEM1,0  
MOVF MEM2,0  
GOTO START  
END
```

MEM1
MEM2

ファイルレジスタ



Wレジスタ



wレジスタの内容
をファイルレジスタ
のMEM2に転送
せよという命令

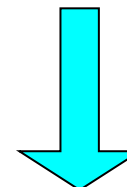
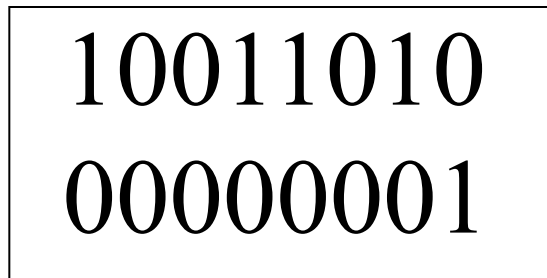
; Data from Working Register to File Register and vice versa

```
INCLUDE "p16F84A.inc"  
list p=16F84A
```

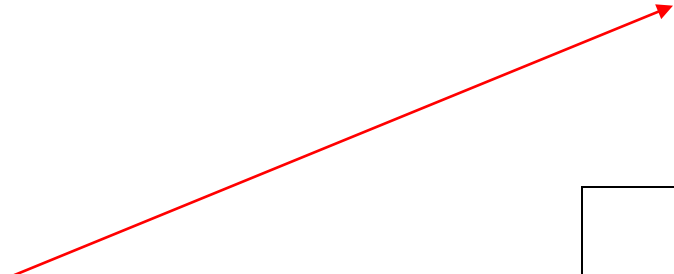
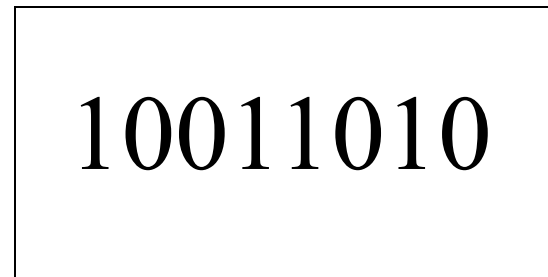
```
__CONFIG _HS_OSC & _WDT_OFF &  
MEM1 EQU 0x0C  
MEM2 EQU 0x0C+1  
ORG 0  
GOTO START  
ORG 4
```

```
START  
MOVLW 0x9A  
MOVWF MEM1  
MOVLW 0x01  
MOVWF MEM2  
MOVF MEM1,0  
MOVF MEM2,0  
GOTO START  
END
```

ファイルレジスタ



Wレジスタ



MEM1の内容をwレジスタに転送せよという命令

MOVF f, d
レジスタfの中身をdへ転送せよという命令
d = 0はWレジスタ

; Data from Working Register to File Register and vice versa

```
INCLUDE "p16F84A.inc"  
list p=16F84A
```

```
__CONFIG _HS_OSC & _WDT_OFF &  
MEM1 EQU 0x0C  
MEM2 EQU 0x0C+1  
  
ORG 0  
GOTO START  
ORG 4
```

START

```
MOVLW 0x9A  
MOVWF MEM1  
MOVLW 0x01  
MOVWF MEM2  
MOVF MEM1,0  
MOVF MEM2,0  
  
GOTO START  
END
```

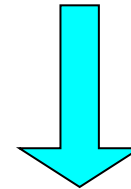
MEM1

MEM2

ファイルレジスタ

10011010

00000001



Wレジスタ

00000001

MEM2の内容をw
レジスタに転送せ
よという命令

; Data from Working Register to File Register and vice versa

```
INCLUDE "p16F84A.inc"  
list p=16F84A
```

```
__CONFIG _HS_OSC & _WDT_OFF &  
MEM1 EQU 0x0C  
MEM2 EQU 0x0C+1  
  
ORG 0  
GOTO START  
ORG 4
```

START

```
MOVLW 0x9A  
MOVWF MEM1  
MOVLW 0x01  
MOVWF MEM2  
MOVF MEM1,0  
MOVF MEM2,0
```

```
GOTO START
```

```
END
```

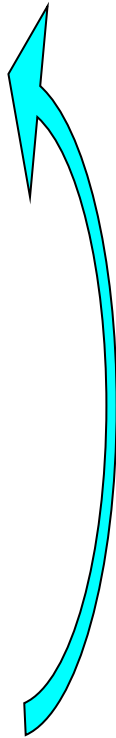
ファイルレジスタ

MEM1	10011010
MEM2	00000001

Wレジスタ

00000001

STARTへジャンプする.



演習問題 1. MEM1に0xAA, MEM2に0xBBを転送し, これらを交換するプログラムを作成せよ.

演習問題 1. MEM1に0xAA, MEM2に0xBBを転送し, これらを交換するプログラムを作成せよ.

; Problem 1

```
INCLUDE"p16F84A.inc"
list p=16F84A

__CONFIG _HS_OSC & _WDT_OFF & _PWRTE_OFF & _CP_OFF

MEM1 EQU 0x0C ;MEM1 at 0C
MEM2 EQU 0x0C+1 ;MEM2 at 0D
MEM3 EQU 0x0C+2 ;MEM3 at 0E

ORG 0
GOTO START
ORG 4

START

MOVLW 0xAA ; 'AA' to Working Register
MOVWF MEM1 ; Move 'AA' to MEM1
MOVLW 0xBB ; 'BB' to Working Register
MOVWF MEM2 ; Move 'BB' to MEM2

MOVF MEM1,0 ;Load MEM1 to W
MOVWF MEM3 ;Move W to MEM3
MOVF MEM2,0 ;Load MEM2 to W
MOVWF MEM1 ;Move W to MEM1
MOVF MEM3,0 ;Load MEM3 to W
MOVWF MEM2 ;Move W to MEM2

GOTO START
END
```

MEM1に0xAA,
MEM2に0xBBを転送

MEM1の内容を一端
MEM3に転送し, MEM2
の内容をMEM1に移した
後に, MEM3の内容を
MEM2に転送

2004年8月

著者： 古橋武
名古屋大学工学研究科計算理工学専攻
furuhashi@cse.nagoya-u.ac.jp