Chip: Simple, V0
Pennsylvania State University

Documented By
David Schor

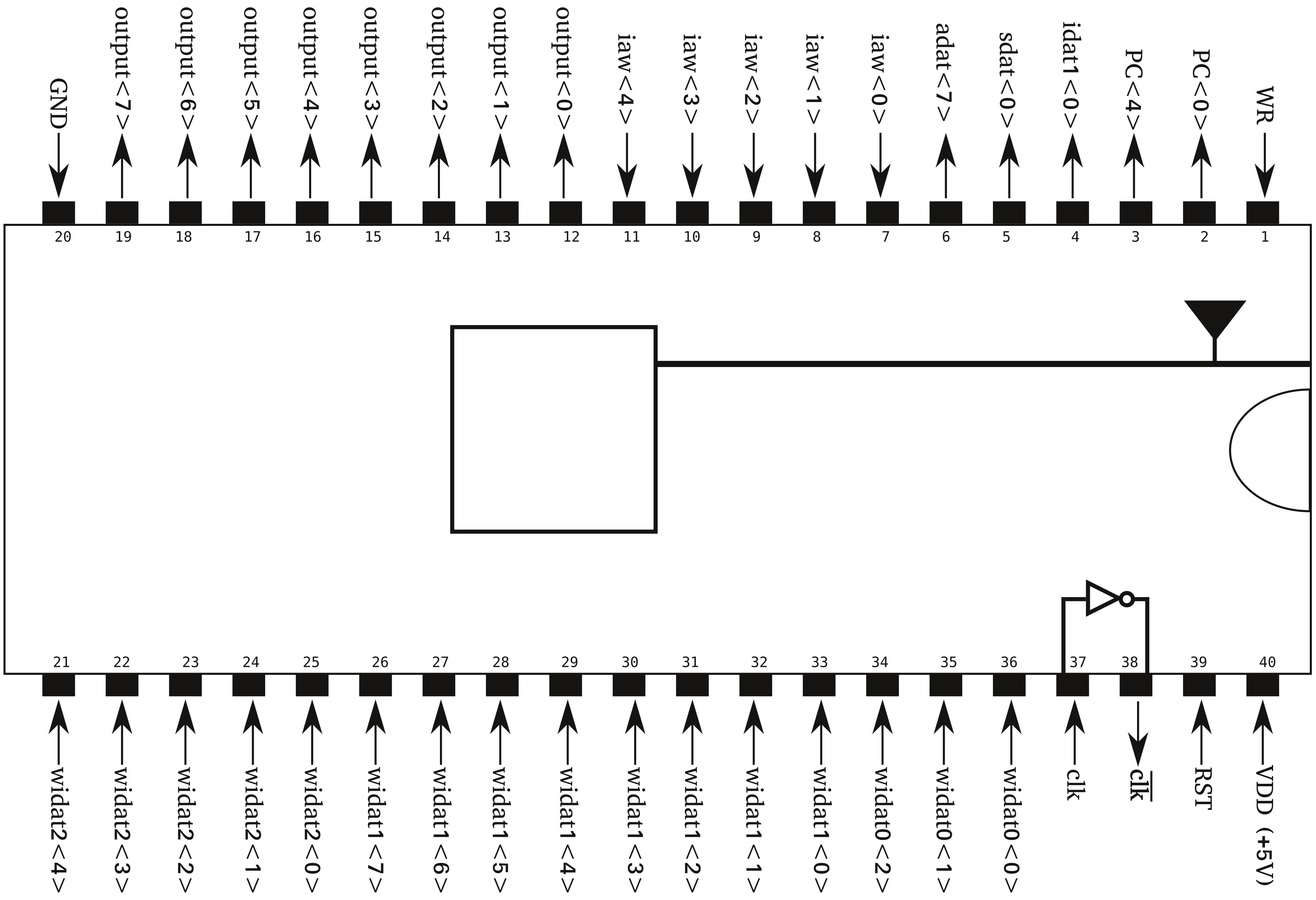
DATA SHEET

Simple, version 0
Chip Specification

Chip: Simple, V0

Pennsylvania State University

Layout By
David Schor



Pin Descriptions:

PIN	NAME	DESCRIPTION
1	WR	Write Enable
2	PC<0>	LSB of the PC
3	PC<4>	5th bit of the program counter
4	idat1<0>	Program's memory data, bit 0
5	sdat<0>	First operand, bit 0
6	adat<7>	ALU's sign bit
7	iaw<0>	Write address, bit 0
8	iaw<1>	Write address, bit 1
9	iaw<2>	Write address, bit 2
10	iaw<3>	Write address, bit 3
11	iaw<4>	Write address, bit 4
12	output<0>	Output port, bit 0
13	output<1>	Output port, bit 1
14	output<2>	Output port, bit 2
15	output<3>	Output port, bit 3
16	output<4>	Output port, bit 4
17	output<5>	Output port, bit 5
18	output<6>	Output port, bit 6
19	output<7>	Output port, bit 7
20	GNT	Ground terminal
21	widat2<4>	Write data, bit 15
22	widat2<3>	Write data, bit 14
23	widat2<2>	Write data, bit 13
24	widat2<1>	Write data, bit 12
25	widat2<0>	Write data, bit 11
26	widat1<7>	Write data, bit 10
27	widat1<6>	Write data, bit 9
28	widat1<5>	Write data, bit 8
29	widat1<4>	Write data, bit 7
30	widat1<3>	Write data, bit 6
31	widat1<2>	Write data, bit 5
32	widat1<1>	Write data, bit 4
33	widat1<0>	Write data, bit 3
34	widat0<2>	Write data, bit 2
35	widat0<1>	Write data, bit 1
36	widat0<0>	Write data, bit 0
37	clk	Clock in
38	$\overline{\text{clk}}$	Clock out, $\overline{\text{clk}}$
39	RST	Reset
40	Vdd	Power terminal

I/O Ports:

I/O Operations are govern by the IN and OUT instructions. The OUT instruction sends the specified register's value to the output ports (output<7:0>, pins 12 to 19). The output port will remain until another OUT instruction overrides it with a new value. The IN instruction reads the input from the input ports (which are duplexed with widat1<7:0>, pins 26 to 33):

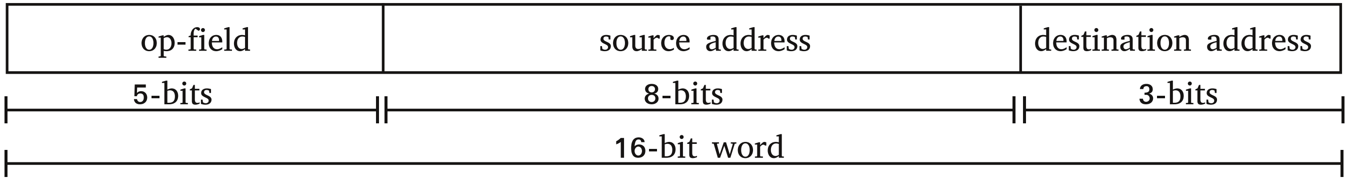
PIN	NAME	DESCRIPTION
26	idat1<7>	Input port, bit 7
27	idat1<6>	Input port, bit 6
28	idat1<5>	Input port, bit 5
29	idat1<4>	Input port, bit 4
30	idat1<3>	Input port, bit 3
31	idat1<2>	Input port, bit 2
32	idat1<1>	Input port, bit 1
33	idat1<0>	Input port, bit 0

Instruction Set

The chip contains two dedicated memory banks: Program Memory (PM) and Data Memory (DM). The program memory is an array of 16-bit words. The data memory is an array of 8-bit words.

instruction Field:

All instructions are 16-bit words. The five most significant bits are the instruction op-code. The middle eight bits (1-byte) is the source address or immedate value. The three least significant bits are for the destination address.



Addressing Modes:

INSTRUCTION	IMMEDIATE	RR
NOP	✘	✔
MV	✔	✔
ADD	✘	✔
SUB	✘	✔
BC	✘	✔
IN	✘	✔
OUT	✘	✔

Instruction Set

The assembler supports all opcode plus a number of pseudoinstructions.

```
 1  MV   immed, reg
 2  MV   reg, reg
 3  ADD  reg, reg
 4  SUB  reg, reg
 5  INC  reg, reg
 6  DEC  reg, reg
 7  DECZ reg, reg
 8  BC   label
 9  BLT  reg, reg, label
10. BGT  reg, reg, label
11. BNZ  reg, reg, label
12. IN   reg
13. OUT  reg
14. NOP  reg
15. CLR  reg
```

MV

Moves data from source to destination

Operation: R_{src} ⇨ R_{dest}

Description: Moves the content of one register to another register. The content of the source register is not changed.

Ctrl Flags:

C
-

Instruction	Adr. Mode	Obj. Code
MV immed, reg	Immediate	00100|imd|dest
MV reg, reg	RegReg	00000|src|dest

NOP

Null operation

Operation: No Operation

Description: This instruction increments the PC and does nothing else. No other registers are affected. NOPs can be used to create CPU-based time delay.

Ctrl Flags:

C
-

Instruction	Adr. Mode	Obj. Code
NOP	-	00000|00000000|000

ADD

Adds two registers

Operation: $R_{src} + R_{dest} \Rightarrow R_{dest}$

Description: Adds the content of the source register to the destination register and saves the result back in the destination register.

A carry occurs when the result cannot fit in the dest register.

Ctrl Flags:

C
Δ

Instruction	Adr. Mode	Obj. Code
ADD reg, reg	RegReg	00010|src|dest

SUB

Subtract from a register

Operation: $R_{dest} - R_{src} \Rightarrow R_{dest}$

Description: Subtracts the source register from the destination register and saves the result back in the destination register.

A carry occurs when the result cannot fit in the dest register.

Ctrl Flags:

C
Δ

Instruction	Adr. Mode	Obj. Code
SUB reg, reg	RegReg	00011|src|dest

IN

Read from input port

Operation: idat1$_{<7:0>}$ ⇨ R$_{dest}$

Description: Copys the value from the input port into the destination register.

Ctrl Flags:

C
-

<table>
<tr><th>Instruction</th><th>Adr. Mode</th><th>Obj. Code</th></tr>
<tr><td>IN reg</td><td>Reg</td><td>01000|00000000|dest</td></tr>
</table>

OUT

Write to output port

Operation: R$_{src}$ ⇨ output$_{<7:0>}$

Description: Copys the value from the source register to the output port.

Ctrl Flags:

C
-

<table>
<tr><th>Instruction</th><th>Adr. Mode</th><th>Obj. Code</th></tr>
<tr><td>OUT reg</td><td>Reg</td><td>01001|src|000</td></tr>
</table>

BC

Branch on carry

Operation: `If C = 1, then (PC) + src ⇨ PC`

Description: Tests the C status bit and branches if C = 1.

Ctrl Flags:

C
-

Instruction	Adr. Mode	Obj. Code
BC rel	Relative	10000|src|000