NILOMAT_STUDIOS

NI_CPU-E series

CPU_E/E2/E2G

SPECS

The CPU_E series is a turing complete 8bit processor.
With 2-4 multi purpose registers and 2 ALU registers
and 65k of rom. It also has 255 bytes of ram.

THE ALU

It consists of an addition subtraction unit
and the comparetors larger than smaller than and equals.
It aslo has the logical operation XOR OR and AND.

THE I/O

It has 4 input and 4 output ports.
The general counter x/y(gcx/gcy) is also exeeble from the outside.
Each I/O port is 8 bit (including gcx/gcy).

INSTRUCTIONS

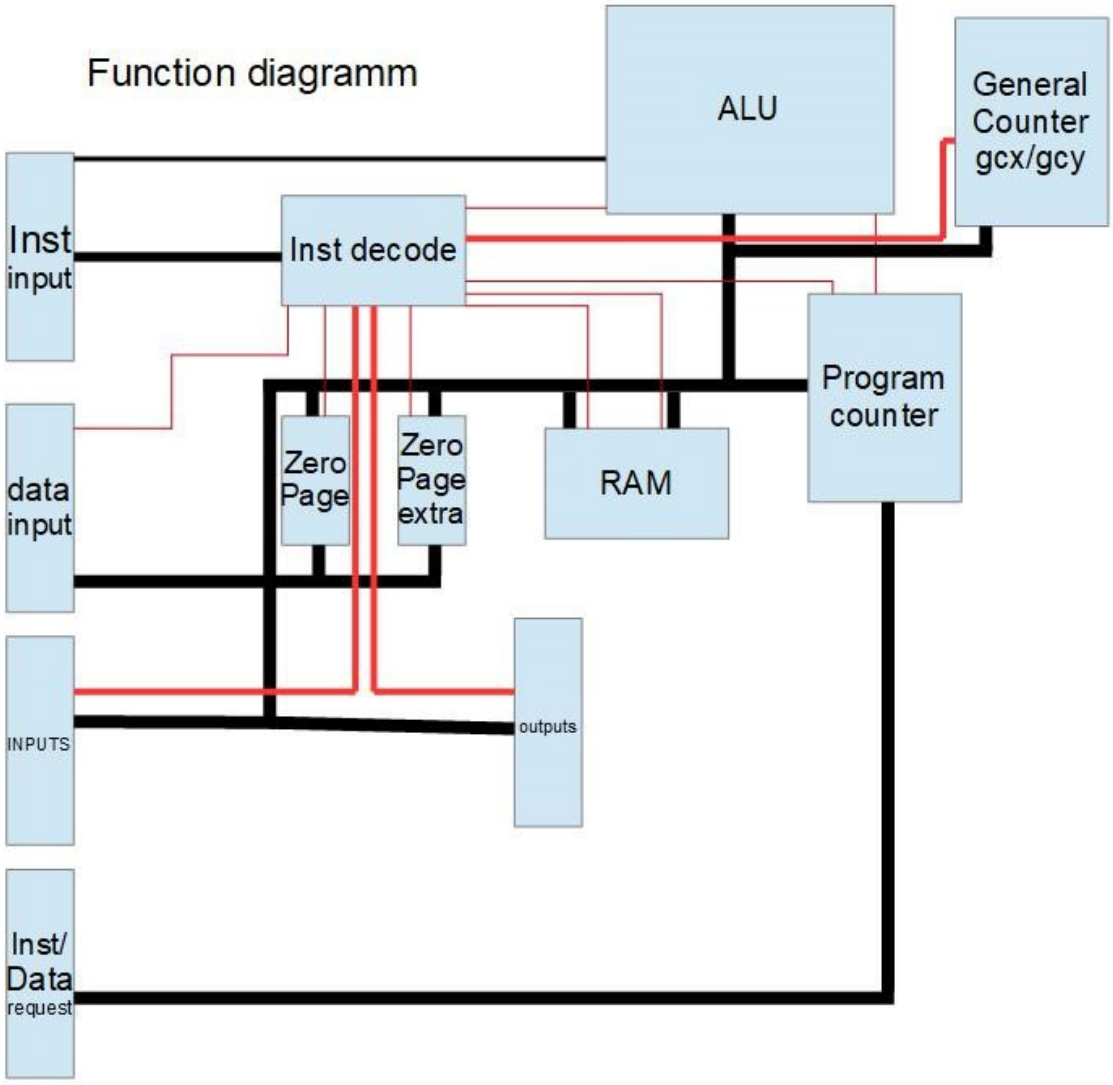
(hex to NISCRIPT)

```
dec hex
31  1F  set ar2
30  1E  set ar1
29  1D  (ALU Operation) *ALU
28  1C  set ram
27  1B  load
26  1A  get ram
25  19  set ra
24  18  output
23  17  set count
22  16  output2
21  15  input
20  14  set zpe
19  13  get zpe
18  12 get count
17  11 set sr
16  10 get sr
15  F  input2
14  E  input3
13  D  input4
12  C  output3
11  B  output4
10  A  inc gcx
9   9  set gcx
8   8  get gcx
7   7  inc gcy
6   6  set gcy
5   5  get gcy
4   4
3   3
2   2
1   1
0   0
```

*ALU
["NAME", hex BYTE]
["add", 1d], ["sub", 5d], ["and", 7d], ["comp", 3d], ["or", 9d], ["xor", bd], ["ar2_gt_ar1", dd], ["ar1_gt_ar2", fd],

LOGIC DIAGRAMM

BLACK LINES – DATA
RED LINES – INSTRUCTION LINES



NEEDED SIGNALS

INPUT

5v
clock
data
instructions(inst)
inputs 1-4

OUTPUT

data 1-4
gcx/gcy
inst data request

NI-SCRIPT BASICS

Command	Description	External Data Needed?
`load <val>`	Load constant into zp	Yes (value)
set ar1`	Move zp to ar1	No
`set ar2`	Move zp to ar2	No
`set ra`	set the ram adress to zp	No
`set ram`	set current ram to zp	No
`set zpe`	set zpe to zp	No
`get ram`	set zp to current ram	No
`get zpe`	set zp to zpe	No
`get input`	set zp to user input	No
`get input2`	set zp to user input 2	No
`get input3`	set zp to user input 3	No
`get input4`	set zp to user input 4	No
`get count`	get the current adress	No
`get sr`	get the mem sector reg	No
`set sr`	set the mem sector reg	No
`add`	Add ar1 + ar2 → zp	No
`sub`	Subtract ar1 - ar2 → zp	No
`and`	Bitwise AND	No
`comp`	jump to zpif ar1==ar2	No
`or`	Bitwise OR	No
`xor`	Bitwise XOR	No
`ar2_gt_ar1`	Compare ar2 > ar1	No
`ar1_gt_ar2`	Compare ar1 > ar2	No
`output`	Output zp	No
`output2`	Output2 zp	No
`output3`	Output3 zp	No
`output4`	Output4 zp	No
`jumpzp`	|jump to value stored in zp	No
`set/get gcy/gcx`	Set/get gcy/gcx	No

IMPORTANT INFO

RA is RAM ADRESS
ZP is ZERO PAGE (ZPE IS ZERO PAGE EXTRA)
AR1/2 is ARITHMATIC REGISTER 1/2
SR is SECTOR REGISTER
GCX/GCY is GENERAL COUNTER X/Y

ALL BITWISE OPERATIONS ALSO USE THE AR REGISTERS

INTEGRATION INFO

You dont have to worry about opcodes,
the 8 bit inst data stream is made of 3 opcode and 5 instruction bits.
The opcode is only used for ALU operations and are listed in the *ALU in the instructions table.

pins

output 1-4
input 1-4
inst data
inst req
run
reset
clk
ram conection out
ram conection in
ram address
ram write
(some extra debug pins might be present)

how to use

if your writing data to the external programm and data memmory set run and reset to low.
And dont forget to disconnect inst req from the data/prog mem for writing.

If you want to halt the processor set run to low and reset to high(will always reset program counter to 0).

the processor will only execute code if the clk is being pulsed.

If you want to restart the processor pulse the reset high for 1-10 clk cycles.
If run is low it will count up in the program but not execute the code.

Behavior

it has 1 instruction per clk cycle.
On reset the prg counter will be set to 0.
If run is off the inst decoder is disabled.

LINKS

NI_CPUE2G COMPILER

NI_CPUE2G DECOMPILER

NI_SCRIPT TUTORIAL