Instruction Set

David Farrell

March 2025

Special registers:

- `sr0` - program kounter `pk`,
- `sr1` - stack pointer `sp`,
- `sr2` - temp A `tmpA`,
- `sr3` - temp B `tmpB`,
- `sr4` - temp B `tmpC`,
- `sr5` - offset register `offs`,
- `sr(8-F)` - interrupt table pointers.

1 Instructions

1.1 Control

- `nop`: `0x00`
- `goto`: `0x01`
- `goto_if_zero`: `0x02`
- `goto_if_nonzero`: `0x03`
- `goto_if_negative`: `0x04`
- `goto_if_nonnegative`: `0x05`
- `goto_if_positive`: `0x06`
- `goto_if_nonpositive`: `0x07`
- `goto_if_equal`: `0x08`
- `goto_if_unequal`: `0x09`
- `goto_if_less`: `0x0a`
- `goto_if_greater`: `0x0b`
- `goto_if_leq`: `0x0c`
- `goto_if_geq`: `0x0d`
- `goto_if_equal`: `0x0e`
- `goto_if_unequal`: `0x0f`
- `goto_if_less`: `0x10`
- `goto_if_greater`: `0x11`
- `goto_if_leq`: `0x12`
- `goto_if_geq`: `0x13`

- goto_if_less_unsgn: 0x14
- goto_if_greater_unsgn: 0x15
- goto_if_leq_unsgn: 0x16
- goto_if_geq_unsgn: 0x17
- goto_if_equal_unsgn: 0x18
- goto_if_unequal_unsgn: 0x19
- goto_if_less_unsgn: 0x1a
- goto_if_greater_unsgn: 0x1b
- goto_if_leq_unsgn: 0x1c
- goto_if_geq_unsgn: 0x1d

1.2 Setting

- swap: 0x1e
- set: 0x1f
- set: 0x20
- set: 0x21
- set: 0x22
- set: 0x23
- set: 0x24
- set: 0x25
- set: 0x26
- set: 0x27
- set: 0x28
- set: 0x29
- set: 0x2a
- set: 0x2b
- set: 0x2c
- set: 0x2d
- set: 0x2e
- set: 0x2f
- set: 0x30
- set: 0x31
- set: 0x32
- set: 0x33
- set: 0x34
- set: 0x35
- set: 0x36
- set: 0x37

- set: 0x38
- set: 0x39
- set: 0x3a
- set: 0x3b
- set: 0x3c

1.3 Arithmetic and Logic Operations

- inc: 0x3d
- dec: 0x3e
- not: 0x3f
- lshift: 0x40
- rshift: 0x41
- inc: 0x42
- dec: 0x43
- not: 0x44
- lshift: 0x45
- rshift: 0x46
- add: 0x47
- sub: 0x48
- mul: 0x49
- div: 0x4a
- rem: 0x4b
- ucmp: 0x4c
- cmp: 0x4d
- and: 0x4e
- or: 0x4f
- xor: 0x50
- shift: 0x51
- add: 0x52
- sub: 0x53
- mul: 0x54
- div: 0x55
- rem: 0x56
- ucmp: 0x57
- cmp: 0x58
- and: 0x59
- or: 0x5a
- xor: 0x5b

- shift: 0x5c
- add: 0x5d
- sub: 0x5e
- mul: 0x5f
- div: 0x60
- rem: 0x61
- ucmp: 0x62
- cmp: 0x63
- and: 0x64
- or: 0x65
- xor: 0x66
- shift: 0x67
- add: 0x68
- sub: 0x69
- mul: 0x6a
- div: 0x6b
- rem: 0x6c
- ucmp: 0x6d
- cmp: 0x6e
- and: 0x6f
- or: 0x70
- xor: 0x71
- shift: 0x72

1.4 The Stack

- push: 0x73
- push: 0x74
- push: 0x75
- push: 0x76
- push: 0x77
- push: 0x78
- push: 0x79
- pop: 0x7a
- pop: 0x7b
- pop: 0x7c
- pop: 0x7d
- pop: 0x7e
- pop: 0x7f
- call: 0x80
- fastcall: 0x81
- return: 0x82

1.5 I/O

- `pinmode_input`: `0x83`
- `pinmode_output`: `0x84`
- `pinmode_input`: `0x85`
- `pinmode_output`: `0x86`
- `set_pin_low`: `0x87`
- `set_pin_high`: `0x88`
- `set_pin_low`: `0x89`
- `set_pin_high`: `0x8a`
- `read_pin`: `0x8b`
- `write_pin`: `0x8c`
- `read_pin`: `0x8d`
- `write_pin`: `0x8e`
- `prints`: `0x8f`
- `prints`: `0x90`
- `syscall`: `0xfd`

1.6 Other

Finally, there is one "inaccessible" instruction: this is used only internally by the control unit, and isn't accepted by the assembler. It is injected whenever an interrupt is detected.

- `handle_interrupt`: `0xfe`