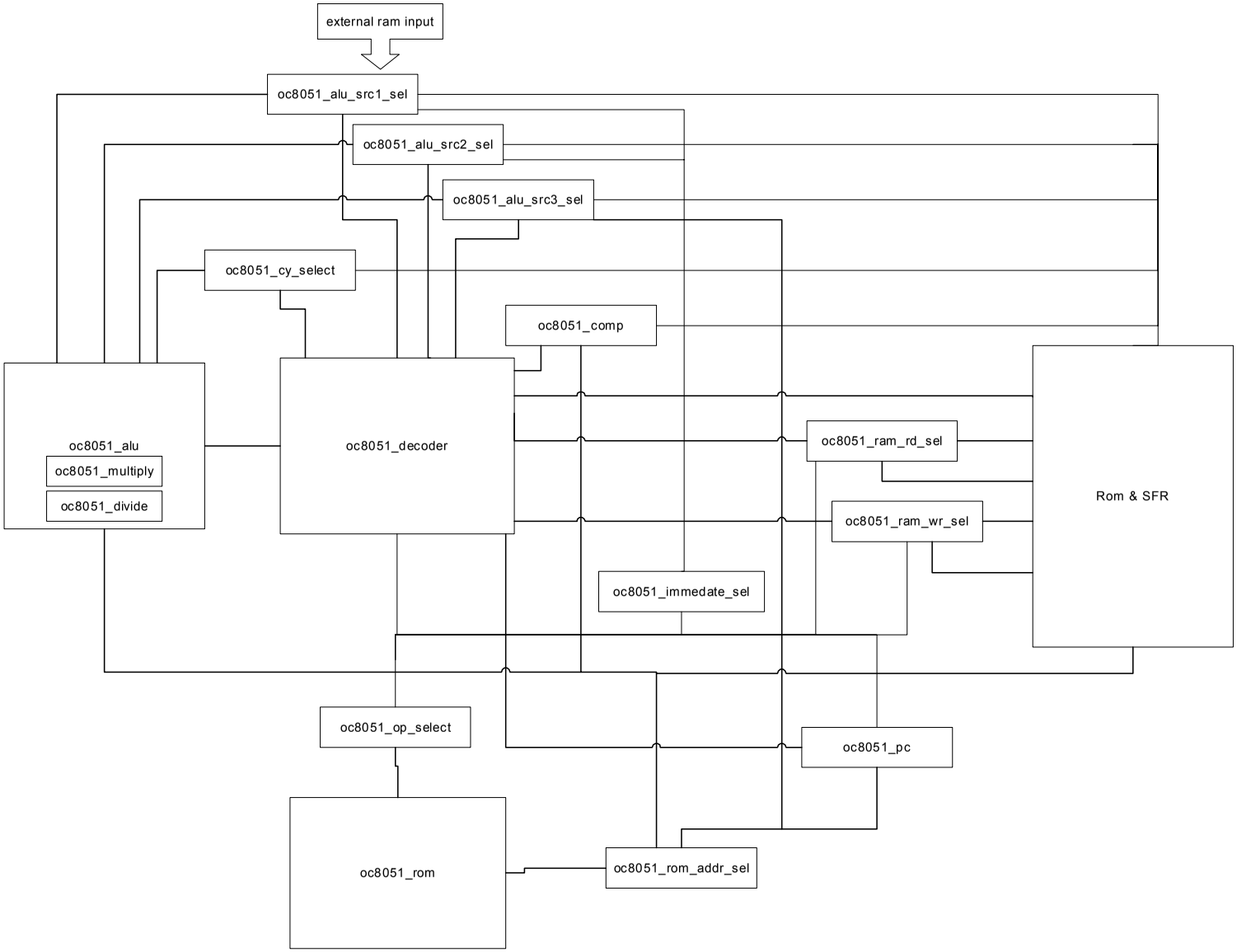

external ram input
oc8051_alu_src1_sel
oc8051_alu_src2_sel
oc8051_alu_src3_sel
oc8051_cy_select
oc8051_comp
oc8051_alu
oc8051_multiply
oc8051_divide
oc8051_decoder
oc8051_ram_rd_sel
oc8051_ram_wr_sel
Rom & SFR
oc8051_immedate_sel
oc8051_op_select
oc8051_pc
oc8051_rom
oc8051_rom_addr_sel

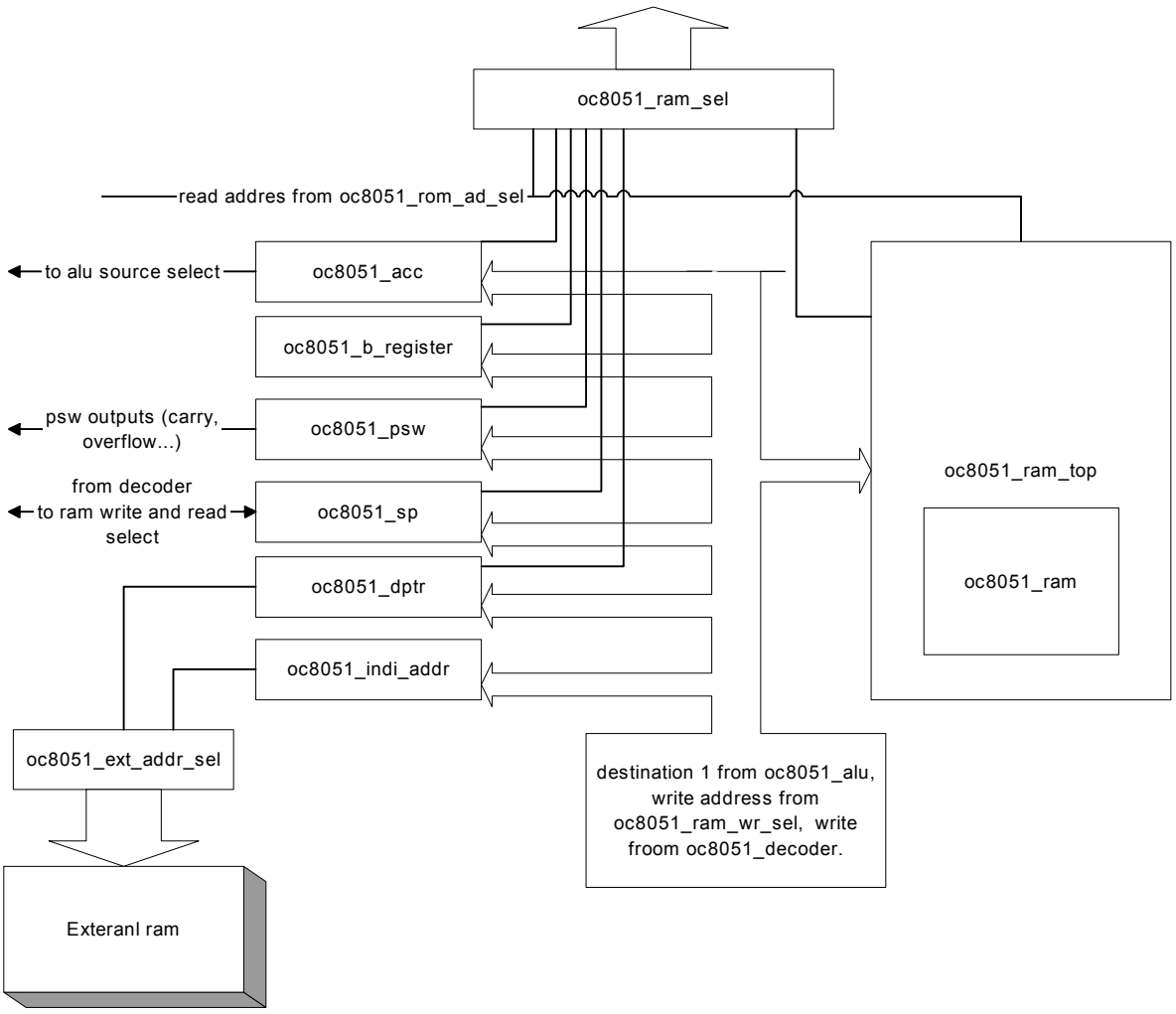
```
module oc8051_top (rst, clk, int, int_v, reti, data_in, data_out, ext_addr,
write, p0_in, p1_in, p2_in, p3_in, p0_out, p1_out, p2_out, p3_out);
//
// rst           (in)  reset - pin
// clk           (in)  clock - pin
// rom_addr      (out) program rom addres (pin + internal)
// int           (in)  interrupt*
// int_v         (in)  interrupt vector*
// reti          (out) return from interrupt*
// data_in       (in)  exteranal ram input
// data_out      (out) exteranal ram output
// ext_addr      (out) external address
// write         (out) write to external ram
// p0_in, p1_in, p2_in, p3_in           (in)  port inputs
// p0_out, p1_out, p2_out, p3_out       (out) port outputs
//
// * in final version this connections will not be pins
//


module oc8051_decoder (clk, rst, op_in, eq, ram_rd_sel, ram_wr_sel, bit_addr,
wr, src_sel1, src_sel2, src_sel3, alu_op, psw_set, cy_sel, imm_sel, pc_wr,
pc_sel,
               comp_sel, rom_addr_sel, ext_addr_sel, wad2, rd, write_x, reti,
rmw);
//
// clk          (in)  clock
// rst          (in)  reset
// op_in        (in)  operation code [oc8051_op_select.op1_out]
// eq           (in)  compare result [oc8051_comp.eq]
// ram_rd_sel   (out) select, whitch address will be send to ram for read
[oc8051_ram_rd_sel.sel, oc8051_sp.ram_rd_sel]
// ram_wr_sel   (out) select, whitch address will be send to ram for write
[oc8051_ram_wr_sel.sel -r, oc8051_sp.ram_wr_sel -r]
// wr           (out) write - if 1 then we will write to ram
[oc8051_ram_top.wr -r, oc8051_acc.wr -r, oc8051_b_register.wr -r, oc8051_sp.wr
-r, oc8051_dptr.wr -r, oc8051_psw.wr -r, oc8051_indi_addr.wr -r,
oc8051_ports.wr -r]
// src_sel1     (out) select alu source 1 [oc8051_alu_src1_sel.sel -r]
// src_sel2     (out) select alu source 2 [oc8051_alu_src2_sel.sel -r]
// src_sel3     (out) select alu source 3 [oc8051_alu_src3_sel.sel -r]
// alu_op       (out) alu operation [oc8051_alu.op_code -r]
// psw_set      (out) will we remember cy, ac, ov from alu [oc8051_psw.set -r]
// cy_sel       (out) carry in alu select [oc8051_cy_select.cy_sel -r]
// comp_sel     (out) compare source select [oc8051_comp.sel]
// bit_addr     (out) if instruction is bit addresable
[oc8051_ram_top.bit_addr -r, oc8051_acc.wr_bit -r, oc8051_b_register.wr_bit -
r, oc8051_sp.wr_bit -r, oc8051_dptr.wr_bit -r, oc8051_psw.wr_bit -r,
oc8051_indi_addr.wr_bit -r, oc8051_ports.wr_bit -r]
// wad2         (out) write acc from destination 2 [oc8051_acc.wad2 -r]
// imm_sel      (out) immediate select [oc8051_immediate_sel.sel -r]
// pc_wr        (out) pc write [oc8051_pc.wr]
// pc_sel       (out) pc select [oc8051_pc.pc_wr_sel]
// rom_addr_sel (out) rom address select (alu destination or pc)
[oc8051_rom_addr_sel.select]
// ext_addr_sel (out) external address select (dptr or Ri)
[oc8051_ext_addr_sel.select]
```

```
// rd           (out) read from rom [oc8051_pc.rd, oc8051_op_select.rd]
// write_x      (out) write to external rom [pin]
// reti         (out) return from interrupt [pin]
// rmw          (out) read modify write feature [oc8051_ports.rmw]
//


module oc8051_alu (op_code, src1, src2, src3, srcCy, srcAc, bit_in, des1,
des2, desCy, desAc, desOv);
//
// op_code      (in)  operation code [oc8051_decoder.alu_op -r]
// src1         (in)  first operand [oc8051_alu_src1_sel.des]
// src2         (in)  second operand [oc8051_alu_src2_sel.des]
// src3         (in)  third operand [oc8051_alu_src3_sel.des]
// srcCy        (in)  carry input [oc8051_cy_select.data_out]
// srcAc        (in)  auxiliary carry input [oc8051_psw.data_out[6] ]
// bit_in       (in)  bit input, used for logic operatins on bits
[oc8051_ram_sel.bit_out]
// des1         (out) first result [oc8051_pc.alu, oc8051_ram_top.wr_data,
oc8051_acc.data_in, oc8051_b_register.data_in, oc8051_comp.des -r,
oc8051_sp.data_in, oc8051_dptr.data_in, oc8051_psw.data_in,
oc8051_indi_addr.data_in, oc8051_rom_addr_sel.des1, oc8051_ports.data_in]
// des2         (out) second result [oc8051_pc.alu, oc8051_rom_addr_sel.des1]
// desCy        (out) carry output [oc8051_ram_top.bit_data_in,
oc8051_acc.bit_in, oc8051_b_register.bit_in, oc8051_psw.cy_in,
oc8051_ports.bit_in]
// desAc        (out) auxiliary carry output [oc8051_psw.ac_in]
// desOv        (out) Overflow output [oc8051_psw.ov_in]
//


module oc8051_divide (src1, src2, des1, des2, desOv);
//
// this module is part of alu
// src1         (in)  first operand
// src2         (in)  second operand
// des1         (out) first result
// des2         (out) second result
// desOv        (out) Overflow output
//


module oc8051_multiply (src1, src2, des1, des2, desOv);
//
// this module is part of alu
// src1         (in)  first operand
// src2         (in)  second operand
// des1         (out) first result
// des2         (out) second result
// desOv        (out) Overflow output
//


module oc8051_alu_src1_sel (sel, immediate, acc, ram, ext, des);
//
// sel          (in)  select signals (from decoder, delayd one clock)
[oc8051_decoder.src_sel1 -r]
```

```
// immediate    (in)  immediate operand [oc8051_immediate_sel.out1]
// acc          (in)  accomulator [oc8051_acc.data_out]
// ram          (in)  ram input [oc8051_ram_sel.out_data]
// ext          (in)  external ram input [pin]
// des          (out) output (alu sorce 1) [oc8051_alu.src1]
//


module oc8051_alu_src2_sel (sel, immediate, acc, ram, des);
//
// sel          (in)  select signals (from decoder, delayd one clock)
[oc8051_decoder.src_sel2 -r]
// immediate    (in)  immediate data [oc8051_immediate_sel.out2]
// acc          (in)  accomulator [oc8051_acc.data_out]
// ram          (in)  ram input [oc8051_ram_sel.out_data]
// des          (out) output (alu sorce 2) [oc8051_alu.src2]
//


module oc8051_alu_src3_sel (sel, pc, dptr, out);
//
// sel          (in)  select signals (from decoder, delayd one clock)
[oc8051_decoder.src_sel3 -r]
// pc           (in)  program counter input [oc8051_pc.pc[15:8] -r]
// dptr         (in)  data pointer input [oc8051_dptr.data_hi]
// des          (out) output (alu sorce 3) [oc8051_alu.src2]
//


module oc8051_comp (sel, b_in, cy, acc, ram, op2, des, eq);
//
// sel          (in)  select whithc sourses to compare (look defines.v)
[oc8051_decoder.comp_sel]
// b_in         (in)  bit in - output from bit addressable memory space
[oc8051_ram_sel.bit_out]
// cy           (in)  carry flag [oc8051_psw.data_out[7] ]
// acc          (in)  accumulator [oc8051_acc.data_out]
// ram          (in)  input from ram [oc8051_ram_sel.out_data]
// op2          (in)  immediate data [oc8051_op_select.op2_out -r]
// des          (in)  destination from alu [oc8051_alu.des1 -r]
// eq           (out) if (src1 == src2) eq = 1  [oc8051_decoder.eq]
//


module oc8051_cy_select (cy_sel, cy_in, data_in, data_out);
//
// cy_sel       (in)  carry select, from decoder (see defines.v)
[oc8051_decoder.cy_sel -r]
// cy_in        (in)  carry input [oc8051_psw.data_out[7] ]
// data_in      (in)  ram data input [oc8051_ram_sel.bit_out]
// data_out     (out) data output [oc8051_alu.srcCy]
//


module oc8051_immediate_sel (sel, op2, op3, pch, pcl, out1, out2);
//
// sel          (in)  select (from decoder) [oc8051_decoder.imm_sel]
```

```
// op2          (in)  byte 2 [oc8051_op_select.op2_out]
// op3          (in)  byte 3 [oc8051_op_select.op3_out]
// pch          (in)  pc high [oc8051_pc.pc_out[15:8] -r]
// pcl          (in)  pc low [oc8051_pc.pc_out[7:0] ]
// out1         (out) output to alu source select 1
[oc8051_alu_src1_sel.immediate]
// out2         (out) output to alu source select 2
[oc8051_alu_src2_sel.immediate]
//


module oc8051_op_select (clk, int, rd, int_v, op1, op2, op3, op1_out, op2_out,
op2_direct, op3_out);
//
// clk          (in)  clock
// int          (in)  interrupt [pin]
// int_v        (in)  interrupt vector (low byte) [pin]
// op1, op2, op3 (in)  input from rom (instruction bytes) [pin]
// rd           (in)  read from rom [oc8051_decoder.rd]
// op1_out      (out) byte 1 output [oc8051_pc.op1, oc8051_decoder.op_in]
// op2_out      (out) byte 2 output [oc8051_pc.op2, oc8051_immediate_sel.op2,
oc8051_comp.op2 -r]
// op2_direct   (out) byte 2 output (used for direct addressing)
[oc8051_ram_rd_sel.imm, oc8051_ram_wr_sel.imm -r]
// op3_out      (out) byte 3 output [oc8051_pc.op3, oc8051_ram_wr_sel.imm2,
oc8051_immediate_sel.op3
//


module oc8051_pc (rst, clk, pc_out, alu, pc_wr_sel, op1, op2, op3, wr, rd,
int);
//
// rst          (in)  reset
// clk          (in)  clock
// pc_out       (out) output, connected to rom_addr_sel1, it's current rom
addres [oc8051_rom_addr_sel.pc]
// alu          (in)  input from alu, used in case of jumps (next addres is
calculated in alu and written to pc) [{oc8051_alu.des1,oc8051_alu.des2}]
// pc_wr_sel    (in)  input indicates whitch input will be written to pc in
case of writting [oc8051_decoder.pc_sel]
// op1          (in)  instruction (byte 1) used to calculate next addres
[oc8051_op_select.op1_out]
// op2          (in)  instruction (byte 2) used in jumps
[oc8051_op_select.op2_out]
// op3          (in)  instruction (byte 3) used in jumps
[oc8051_op_select.op3_out]
// wr           (in)  write (active high) [oc8051_decoder.pc_wr]
// rd           (in)  read: if high calculate next addres else hold current
[oc8051_decoder.rd]
// int          (in)  interrupt (if high don't change outputs -- write to
stack) [pin]
//


module oc8051_ram_rd_sel (sel, sp, ri, rn, imm, out);
//
// sel          (in)  select (look defines) [oc8051_decoder.ram_rd_sel]
```

```
// sp           (in)  stack ponter [oc8051_sp.data_out]
// ri           (in)  indirect addresing [oc8051_indi_addr.data_out]
// rn           (in)  registers [{oc8051_psw.data_out[4:3],
oc8051_op_select.op1_out[2:0]}]
// imm          (in)  immediate (direct addresing)
[oc8051_op_select.op2_direct]
// out          (out) output [oc8051_ram_top.rd_addr, oc8051_ram_sel.addr -r,
oc8051_ports.rd_addr]
//


module oc8051_ram_wr_sel (sel, sp, rn, imm, ri, imm2, out);
//
// sel          (in)  select (look defines) [oc8051_decoder.ram_wr_sel -r]
// sp           (in)  stack ponter [oc8051_sp.data_out]
// ri           (in)  indirect addressing [oc8051_indi_addr.data_out -r]
// rn           (in)  registers [{oc8051_psw.data_out[4:3],
oc8051_op_select.op1_out[2:0]} -r]
// imm          (in)  immediate, byte 2 (direct addresing)
[oc8051_op_select.op2_direct -r]
// imm2         (in)  immediate, byte 3 (direct addresing)
[oc8051_op_select.op3_out]
// out          (out) output [oc8051_ram_top.wr_addr, oc8051_acc.wr_addr,
oc8051_b_register.wr_addr, oc8051_sp.wr_addr, oc8051_dptr.addr,
oc8051_psw.addr, oc8051_indi_addr.addr, oc8051_ports.wr_addr]
//


module oc8051_rom_addr_sel (clk, rst, select, des1, des2, pc, op1, out_data,
out_addr);
//
// clk          (in)  clock
// rst          (in)  reset
// select       (in)  output select [oc8051_decoder.rom_addr_sel]
// des1, des2   (in)  alu destination input
[{oc8051_alu.des1,oc8051_alu.des2}]
// pc           (in)  pc input [oc8051_pc.pc_out]
// op1          (in)  byte 1 from rom [pin]
// out_data     (out) output data (alu des2) [oc8051_acc.data2_in,
oc8051_dptr.data2_in]
// out_addr     (out) output address (to program rom) [oc8051_rom.addr]
//


module oc8051_rom (rst, clk, addr, data1, data2, data3);
//
// module is tehnology dependent
//
// clk          (in)  clock
// rst          (in)  reset
// addr         (in)  addres [oc8051_rom_addr_sel.out_addr]
// data1        (out) 1 byte of instruction [oc8051_op_select.op1,
oc8051_rom_addr_sel.op1]
// data2        (out) 2 byte of instruction [oc8051_op_select.op2]
// data3        (out) 3 byte of instruction [oc8051_op_select.op3]
//
```




```
module oc8051_ram_top (clk, rst, rd_addr, rd_data, wr_addr, bit_addr, wr_data,
wr, bit_data_in, bit_data_out);
//
// clk          (in)  clock
// rd_addr      (in)  read addres [oc8051_ram_rd_sel.out]
// rd_data      (out) read data [oc8051_ram_sel.in_ram]
// wr_addr      (in)  write addres [oc8051_ram_wr_sel.out]
// bit_addr     (in)  bit addresable instruction [oc8051_decoder.bit_addr -r]
// wr_data      (in)  write data [oc8051_alu.des1]
// wr           (in)  write [oc8051_decoder.wr -r]
// bit_data_in  (in)  bit data input [oc8051_alu.desCy]
// bit_data_out (out)  bit data output [oc8051_ram_sel.bit_in]
//


module oc8051_ram (clk, rst, rd_addr, rd_data, wr_addr, wr_data, wr);
//
// this module is part of oc8051_ram_top
// it's tehnology dependent
//
// clk          (in)  clock
// rd_addr      (in)  read addres
```

```
// rd_data      (out) read data
// wr_addr      (in)  write addres
// wr_data      (in)  write data
// wr           (in)  write
//


module oc8051_ram_sel (addr, bit_in, in_ram, psw, acc, dptr_hi, ports_in, sp,
b_reg, bit_out, out_data);
//
// addr         (in)  address [oc8051_ram_rd_sel.out -r]
// bit_in       (in)  bit input (from ram) [oc8051_ram_top.bit_data_out]
// in_ram       (in)  imput from ram [oc8051_ram_top.rd_data]
// psw          (in)  program status word input [oc8051_psw.data_out]
// acc          (in)  accumulator input [oc8051_acc.data_out]
// dptr_hi      (in)  data pointer high bits input [oc8051_dptr.data_hi]
// ports_in     (in)  ports input [oc8051_ports.data_out]
// sp           (in)  stack pointer [oc8051_sp.data_out]
// b_reg        (in)  b register input [oc8051_b_register.data_out]
// bit_out      (out) bit output [oc8051_alu.bit_in, oc8051_comp.b_in,
oc8051_cy_select.data_in]
// out_data     (out) data (byte) output [oc8051_alu_src1_sel.ram,
oc8051_alu_src2_sel.ram, oc8051_comp.ram]
//


module oc8051_acc (clk, rst, bit_in, data_in, data2_in, wr, wr_bit, wad2,
wr_addr,
data_out, p);
//
// clk          (in)  clock
// rst          (in)  reset
// bit_in       (in)  bit input - used in case of writing bits to acc (bit
adddressable memory space - alu carry) [oc8051_alu.desCy]
// data_in      (in)  data input - used to write to acc (from alu destiantion
1) [oc8051_alu.des1]
// data2_in     (in)  data 2 input - write to acc, from alu detination 2 -
instuctions mul and div [oc8051_alu.des2]
// wr           (in)  write - actine high [oc8051_decoder.wr -r]
// wr_bit       (in)  write bit addresable - actine high
[oc8051_decoder.bit_addr -r]
// wad2         (in)  write data 2 [oc8051_decoder.wad2 -r]
// wr_addr      (in)  write address (if is addres of acc and white high must
be written to acc) [oc8051_ram_wr_sel.out]
// data_out     (out) data output [oc8051_alu_src1_sel.acc
oc8051_alu_src2_sel.acc oc8051_comp.acc oc8051_ram_sel.acc]
// p            (out) parity [oc8051_psw.p]
//


module oc8051_b_register (clk, rst, bit_in, data_in, wr, wr_bit, wr_addr,
data_out);
//
// clk          (in)  clock
// rst          (in)  reset
// bit_in       (in)  bit input - used in case of writing bits to b register
(bit adddressable memory space - alu carry) [oc8051_alu.desCy]
```

```
// data_in      (in)  data input - used to write to b register
[oc8051_alu.des1]
// wr           (in)  write - actine high [oc8051_decoder.wr -r]
// wr_bit       (in)  write bit addresable - actine high
[oc8051_decoder.bit_addr -r]
// wr_addr      (in)  write address [oc8051_ram_wr_sel.out]
// data_out     (out) data output [oc8051_ram_sel.b_reg]
//


module oc8051_dptr(clk, rst, addr, data_in, data2_in, wr, wd2, wr_bit,
data_hi, data_lo);
//
// clk          (in)  clock
// rst          (in)  reset
// addr         (in)  write address input [oc8051_ram_wr_sel.out]
// data_in      (in)  destination 1 from alu [oc8051_alu.des1]
// data2_in     (in)  destination 2 from alu [oc8051_alu.des2]
// wr           (in)  write to ram [oc8051_decoder.wr -r]
// wd2          (in)  write from destination 2 [oc8051_decoder.ram_wr_sel -r]
// wr_bit       (in)  write bit addresable [oc8051_decoder.bit_addr -r]
// data_hi      (out) output (high bits) [oc8051_alu_src3_sel.dptr,
oc8051_ext_addr_sel.dptr_hi, oc8051_ram_sel.dptr_hi]
// data_lo      (out) output (low bits) [oc8051_ext_addr_sel.dptr_lo]
//


module oc8051_ext_addr_sel (clk, select, write, dptr_hi, dptr_lo, ri,
addr_out);
//
// clk  clock
// select       (in)  select sourses [oc8051_decoder.ext_addr_sel -r]
// write        (in)  write to external ram [oc8051_decoder.write_x]
// dptr_hi      (in)  data pointer high bits [oc8051_dptr.data_hi]
// dptr_lo      (in)  data pointer low bits [oc8051_dptr.data_lo]
// ri           (in)  indirect addressing [oc8051_indi_addr.data_out]
// addr_out     (out) external addres [pin]
//


module oc8051_indi_addr (clk, rst, addr, data_in, wr, wr_bit, data_out, sel,
bank);
//
// clk          (in)  clock
// rst          (in)  reset
// addr         (in)  write address [oc8051_ram_wr_sel.out]
// data_in      (in)  data input (alu destination1) [oc8051_alu.des1]
// wr           (in)  write [oc8051_decoder.wr -r]
// wr_bit       (in)  write bit addresable [oc8051_decoder.bit_addr -r]
// data_out     (out) data output [oc8051_ram_rd_sel.ri, oc8051_ram_wr_sel.ri
-r]
// sel          (in)  select register [oc8051_op_select.op1_out[0] ]
// bank         (in)  select register bank: [oc8051_psw.data_out[4:3] ]
//
```

```
module oc8051_ports (clk, rst, bit_in, data_in, wr, wr_bit, wr_addr, rd_addr,
rmw, data_out, p0_out, p1_out, p2_out, p3_out,
                     p0_in, p1_in, p2_in, p3_in);
//
// clk          (in)  clock
// rst          (in)  reset
// bit_in       (in)  bit input [oc8051_alu.desCy]
// data_in      (in)  data input (from alu destiantion 1) [oc8051_alu.des1]
// wr           (in)  write [oc8051_decoder.wr -r]
// wr_bit       (in)  write bit addresable [oc8051_decoder.bit_addr -r]
// wr_addr      (in)  write address [oc8051_ram_wr_sel.out]
// rd_addr      (in)  read address [oc8051_ram_rd_sel.out]
// rmw          (in)  read modify write feature [oc8051_decoder.rmw]
// data_out     (out) data output [oc8051_ram_sel.ports_in]
// p0_out, p1_out, p2_out, p3_out       (out) port outputs [pin]
// p0_in, p1_in, p2_in, p3_in           (in)  port inputs [pin]
//


module oc8051_psw (clk, rst, addr, data_in, wr, wr_bit, data_out, p, cy_in,
ac_in, ov_in, set);
//
// clk          (in)  clock
// rst          (in)  reset
// addr         (in)  write address [oc8051_ram_wr_sel.out]
// data_in      (in)  data input [oc8051_alu.des1]
// wr           (in)  write [oc8051_decoder.wr -r]
// wr_bit       (in)  write bit addresable [oc8051_decoder.bit_addr -r]
// data_out     (out) data output [oc8051_ram_sel.psw]
// p            (in)  parity [oc8051_acc.p]
// cy_in        (in)  input bit data [oc8051_alu.desCy]
// ac_in        (in)  auxiliary carry input [oc8051_alu.desAc]
// ov_in        (in)  overflov input [oc8051_alu.desOv]
// set          (in)  set psw (write to caryy, carry and overflov or carry,
owerflov and ac) [oc8051_decoder.psw_set -r]
//


module oc8051_sp (clk, rst, ram_rd_sel, ram_wr_sel, wr_addr, wr, wr_bit,
data_in, data_out);
//
// clk          (in)  clock
// rst          (in)  reset
// ram_rd_sel   (in)  ram read select, used tu calculate next value
[oc8051_decoder.ram_rd_sel]
// ram_wr_sel   (in)  ram write select, used tu calculate next value
[oc8051_decoder.ram_wr_sel -r]
// wr           (in)  write [oc8051_decoder.wr -r]
// wr_bit       (in)  write bit addresable [oc8051_decoder.bit_addr -r]
// data_in      (in)  data input [oc8051_alu.des1]
// wr_addr      (in)  write address (if is addres of sp and white high must be
written to sp)  [oc8051_ram_wr_sel.out]
// data_out     (out) data output [oc8051_ram_rd_sel.sp, oc8051_ram_rd_sel
oc8051_ram_wr_sel1.sp, oc8051_ram_sel.sp]
//
```