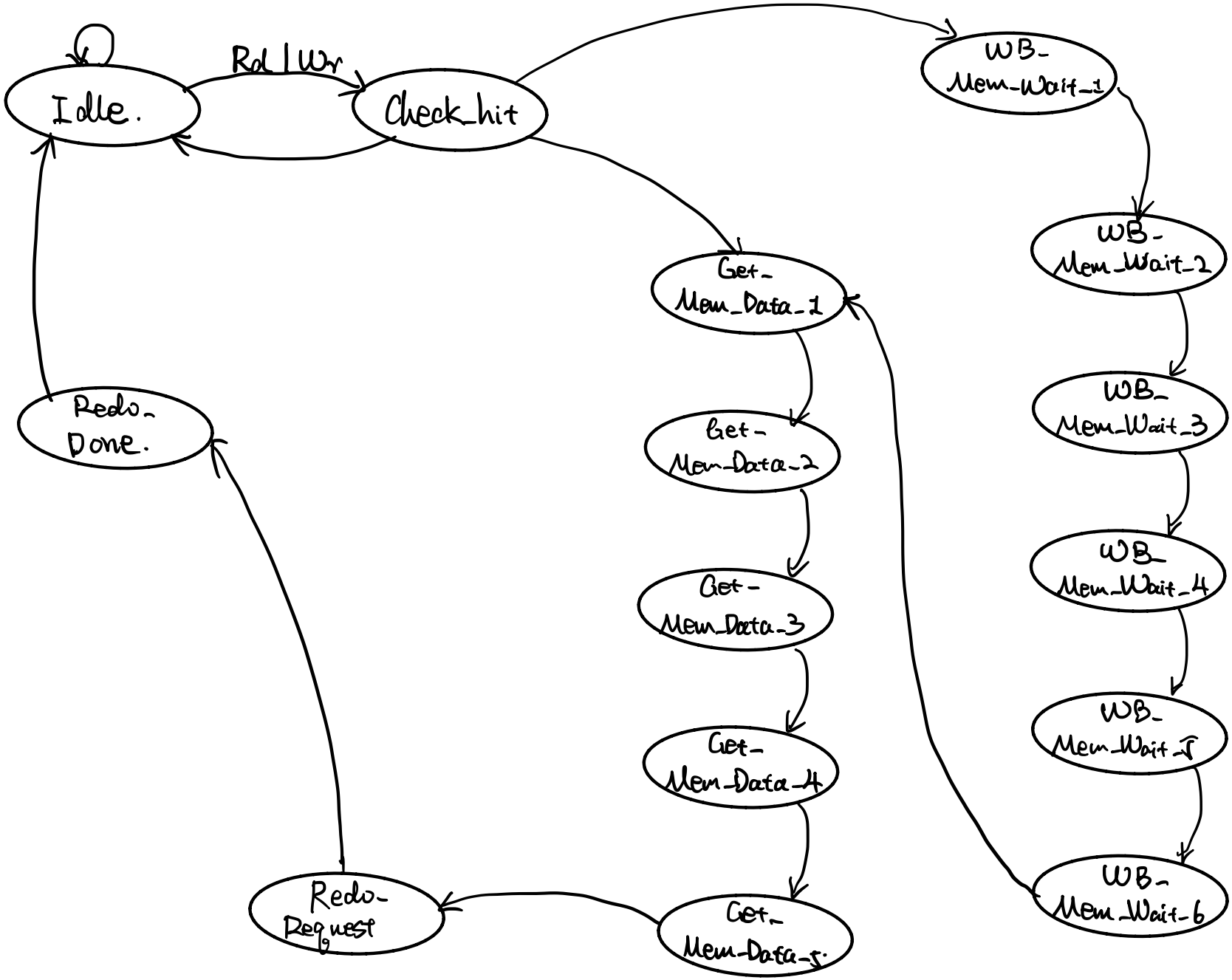
Direct_mapped cache FSM.



Direct_mapped cache FSM Detail

cache_enable = 1’b0
cache_comp= 1’b0
cache_wr = 1’b0
cache_valid_in = 1’b1
cache_data_in + DataIn_reg
cache_tag_in = Addr_reg[15:11]
cache_data_in = Addr_reg[10:3]
cache_offset = Addr_reg[2:0]
mem_addr = Addr_reg
mem_data_in = cache_data_out
mem_wr = 1’b0
mem_rd = 1’b0
fsm_err = 1’b0
fsm_stall = 1’b1
fsm_hit = 1’b0
fsm_done = 1’b0

```
IDLE:
        fsm_err = Rd && Wr
        cache_enable = Rd ^Wr
        cache_comp = Rd^Wr
        cache_data_in = DataIn
        cache_wr = (~Rd)&Wr
        cache_tag_in = Addr[15:11]
        cache_index = Addr[10:3]
        cache_offset = Addr[2:0]
        fsm_stall = 1’b0
        next state:
                if (Rd | Wr):
                        CHECK_HIT
                else:
                         IDLE

CHECK_HIT:
        fsm_done = cahce_hit & cache_valid
        fsm_hit = cache_hit & cache_valid
        if (~cache_hit & cache_valid & cache_dirty):
                mem_data_in = cache_data_out
        else:
                mem_data_in = DataIn_reg
        if (~ cache_hit & cache_valid & cache_dirty):
```

```
            mem_addr = {cache_tag, cache_index, 3'b000}
        elif (cache_hit & ~cache_valid | ~cache_hit & ~cache_drty):
            mem_addr  = {cache_tag_in, cache_index, 3'b000}
        else:
            mem_addr  = Addr_reg
        if (~cache_hit & cache_valid & cache_dirty):
            cache_offset = 3'b000
        else:
            cache_offset  = Addr_reg[2:0]
        cache_enable = 1'b1
        if (~cache_hit & cache_valid & cache_dirty):
            mem_wr = 1'b1
        else:
            mem_wr = 1'b0
        if ((cache_hit & ~cache_valid) | (~cache_hit & ~cache_dirty)):
            mem_rd = 1'b1
        else:
            mem_rd  = 1'b0

        next state:
            if "cache_hit & ~cache_valid " or "~cachhit & ~cache_dirty" :
                GET_MEM_DATA_1
            elif "~cache_hit & cache_valid & cache_dirty" :
                WRITE_BACK_MEM_WAIT_1
            else:
                IDLE
GET_MEM_DATA_1:
        mem_rd = 1'b1
        mem_addr = {cache_tag_in, cache_index, 3'b010}
        next state:
            GET_MEM_DATA_2
GET_MEM_DATA_2:
        mem_rd = 1'b1
        mem_addr = {cache_tag_in, cache_index, 3'b100}
        cache_offset = 3'b000
        cache_enable = 1'b1
        cache_comp = 1'b0
        cache_wr = 1'b1
        cache_valid_in = 1'b1
        cache_data_in = mem_DataOut
        next state:
            GET_MEM_DATA_3
GET_MEM_DATA_3:
```

```
        mem_rd = 1'b1
        mem_addr = {cache_tag_in, cache_index, 3'b110}
        cache_offset = 3'b010
        cache_enable = 1'b1
        cache_comp = 1'b0
        cache_wr = 1'b1
        cache_valid_in = 1'b1
        cache_data_in = mem_DataOut
        next state:
                GET_MEM_DATA_4
GET_MEM_DATA_4:
        cache_offset = 3'b100
        cache_enable = 1'b1
        cache_comp = 1'b0
        cache_wr = 1'b1
        cache_valid_in = 1'b1
        cache_data_in = mem_DataOut
        next state:
                GET_MEM_DATA_5
GET_MEM_DATA_5:
        cache_offset = 3'b110
        cache_enable = 1'b1
        cache_comp = 1'b0
        cache_wr = 1'b1
        cache_valid_in = 1'b1
        cache_data_in = mem_DataOut
        next state:
                REDO_REQUEST
REDO_REQUEST:
        fsm_err = Rd_reg & Wr_reg
        cache_enable = Rd_reg ^ Wr_reg
        cache_comp = Rd_reg^Wr_reg
        cache_wr = (~Rd_reg)&Wr_reg
        next state:
                REDO_DONE
REDO_DONE:
        fsm_stall = 1'b1
        cache_enable = 1'b1
        fsm_done = a'b1
        next state:
                IDLE

WEITE_BACK_MEM_WAIT_1:
```

```
        cache_enable = 1'b1
        cache_wr = 1'b0
        cache_comp = 1'b0
        cache_offset = 3'b010
        mem_wr = 1'b1
        mem_addr = {cache_tag_out, cache_index, 3'b010}
        next state:
                WRITE_BACK_MEM_WAIT_2
WRITE_BACK_MEM_WAIT_2:
        cache_enable = 1'b1
        cache_wr = 1'b0
        cache_comp = 1'b0
        cache_offset = 3'b100
        mem_wr = 1'b1
        mem_addr = {cache_tag_out, cache_index, 3'b100}
        next state:
                WRITE_BACK_MEM_WAIT_3
WRITE_BACK_MEM_WAIT_3:
        cache_enable = 1'b1
        cache_wr = 1'b0
        cache_comp = 1'b0
        cache_offset = 3'b110
        mem_wr = 1'b1
        mem_addr = {cache_tag_out, cache_index, 3'b110}
        next state:
                WRITE_BACK_MEM_WAIT_4
WRITE_BACK_MEM_WAIT_4:
        next_state
                WRITE_BACK_MEM_WAIT_5
WRITE_BACK_MEM_WAIT_5:
        next_state
                WRITE_BACK_MEM_WAIT_6
WRITE_BACK_MEM_WAIT_6:
        mem_addr = {cache_tag_in, cache_index, 3'b000}
        mem_rd = 1'b1
        next state:
                GET_MEM_DATA_1
default:
        fsm_err = 1'b1
```