

1 文件结构图如下



2 主文件 gx18b20_ow_master 的接口使用说明

```
.clk(clk4M), // clock (4MHz +/-5%)
.rst_n(rst_n_r), // asynchronous reset 0 is reset

//one wire signal
ow(ow), //

//user interface input for the module
.ow_start(ow_start), //user interface enable signal;when ow_start=1, signals below is valid for the module. ow_start can not be toggled
.rom_cmd(rom_cmd), //rom_cmd, eg: rom_cmd = 8'hC for ROM_SKIP
.rom_addr(rom_addr), //rom addr, useful with ROM_MATCH
.OWin_cmd(OWin_cmd), // ow_cmd input, eg: OWin_cmd = 8'h0F for Write Scratchpad
.OWin_catch_data(OWin_catch_data), //8 bytes data for scratchpad

//user interface output for the module
.busy(busy), //the module is in busy when OWout_finish=0.
.OWout_finish(OWout_finish), //user operation finished signal.valid when risingedge .Signals below is valid when OWout_finish's rising edge
.OWout_rom_addr(OWout_rom_addr), //rom address-out for rom read command;
.OWout_catch_data(OWout_catch_data), //8 bytes data from scratchpad

.OWout_cmd(OWout_cmd), //current ow cmd output ;
.OW_data_read(OW_data_read), // data read output for any command that including read data. only valid when OW_data_read_valid=1
.OW_data_read_valid(OW_data_read_valid), // valid signal output for OW_data_read
.OW_data_read_cnt(OW_data_read_cnt), // debug output-ports, not for users

//rom search signals outputs
.rom_ID(rom_ID), //only valid when rom_ID_valid=1
.rom_ID_cnt(rom_ID_cnt), //the count of romid only valid when rom_ID_valid=1
.rom_ID_valid(rom_ID_valid), //
.err_id_remove(err_id_remove),
.flag_rom_search_over(flag_rom_search_over),
//error signals outputs when OWout_finish's rising edge
.OWout_dataCRC_err(OWout_dataCRC_err) //read data crc error. for RSP RAP ARAP or RFS command
```

连接GX18b20的芯片接口

用户的指令输入口, 包括rom指令 和对应的rom地址 1wire指令 和对应的数据等. 当ow_start为高时有效. ow_start一个时钟高电平即可, 不可连续输入指令.

当OWout_finish为高时, 表明操作完成, 并且OWout_catch_data就是输出的缓存数据, 一共8个byte. 不包括crc. 如果crc出错, 会在下面的OWout_dataCRC_err拉高. 当rom read指令时, OWout_rom_addr会返回读到的romaddr

测试用, 用户不用

rom_search指令的返回值

3 实例说明 位于文件 gx18b20_master_fpga_top 内, 从 239 行开始

```
237 else if(key_press_cnt==0)begin
238   if(key1_press)begin
239     ow_start ≤ 1 ;
240     rom_cmd ≤ ROM_SEARCH;
241     rom_addr ≤ 0;
242     OWin_cmd ≤ 0 ;
243     OWin_catch_data ≤ 0 ;
244   end
245   else if(key2_press)begin
246     ow_start ≤ 1 ;
247     rom_cmd ≤ ROM_MATCH;
248     rom_addr ≤ rom_addr_1;
249     OWin_cmd ≤ CMD_WSP ;
250     OWin_catch_data ≤ 40'h100c7f0408 ;
251   end
252   else if(key3_press)begin
253     ow_start ≤ 1 ;
254     rom_cmd ≤ ROM_MATCH;
255     rom_addr ≤ rom_addr_1;
256     OWin_cmd ≤ CMD_RSP ;
257     OWin_catch_data ≤ 40'h100c7f0408 ;
258   end
259   else if(key4_press)begin
260     ow_start ≤ 1 ;
261     rom_cmd ≤ ROM_MATCH;
262     rom_addr ≤ rom_addr_1;
263     OWin_cmd ≤ CMD_CSP ;
264     OWin_catch_data ≤ 40'h100c7f0408 ;
265   end
266   else begin
267     ow_start ≤ 0 ;
```

rom_search指令

rom_match指令外加 1wire 的写缓存指令

rom_match指令外加 1wire 的读缓存指令

rom_match指令外加 1wire 的将缓存写入 eeprom指令

```

274   else begin
275     if(key1_press)begin
276       ow_start ≤ 1 ;
277       rom_cmd ≤ ROM_MATCH;
278       rom_addr ≤ rom_addr_1;
279       OWin_cmd ≤ CMD_CONV;
280       OWin_catch_data ≤ 0 ;
281     end
282     else if(key2_press)begin
283       ow_start ≤ 1 ;
284       rom_cmd ≤ ROM_MATCH;
285       rom_addr ≤ rom_addr_1;
286       OWin_cmd ≤ CMD_RSP ;
287       OWin_catch_data ≤ 40'h100c7f0408 ;
288     end
289     else if(key3_press)begin
290       ow_start ≤ 1 ;
291       rom_cmd ≤ ROM_MATCH;
292       rom_addr ≤ rom_addr_1;
293       OWin_cmd ≤ CMD_RSP ;
294       OWin_catch_data ≤ 40'h100c7f0408 ;
295     end
296     else if(key4_press)begin
297       ow_start ≤ 1 ;
298       rom_cmd ≤ ROM_MATCH;
299       rom_addr ≤ rom_addr_1;
300       OWin_cmd ≤ CMD_RSP ;
301       OWin_catch_data ≤ 40'h100c7f0408 ;
302     end

```

rom_match指令，并执行温度转换指令

等待1秒后，rom_match指令并执行读缓存，缓存中有刚才温度转换指令的温度值

```

325
326   always @ (posedge clk4M or negedge rst_n_r)
327   begin
328     if(~rst_n_r)begin
329       rom_addr_1 ≤ 0 ;
330     end
331     else if((rom_ID_cnt == 1) & rom_ID_valid)
332       rom_addr_1 ≤ rom_ID;
333     end
334   always @ (posedge clk4M or negedge rst_n_r)
335   begin
336     if(~rst_n_r)begin
337       rom_addr_2 ≤ 0 ;
338     end
339     else if((rom_ID_cnt == 2) & rom_ID_valid)
340       rom_addr_2 ≤ rom_ID;
341     end
342

```

执行romsearch指令时，当rom_ID_valid输出高时，rom_ID就是搜索到的rom_addr,同时rom_ID_cnt就是搜索到的芯片数量，当搜索完最后一个后，就会停止。这里时保存了搜索到的第一个gx18b20芯片，并将rom_ID保存到rom_addr_1这个寄存器中，用于后续的rom_match操作使用。