SoftReset Documentation

-- HongjiangCai

Instrumentation meeting

Aug 18, 2023

Agenda

- Why we need soft reset?
- How is soft reset achieved?
- How to use soft reset with executable?
- How to use soft reset with inline code?
- How to add soft reset to other modules in the future?

Why soft reset

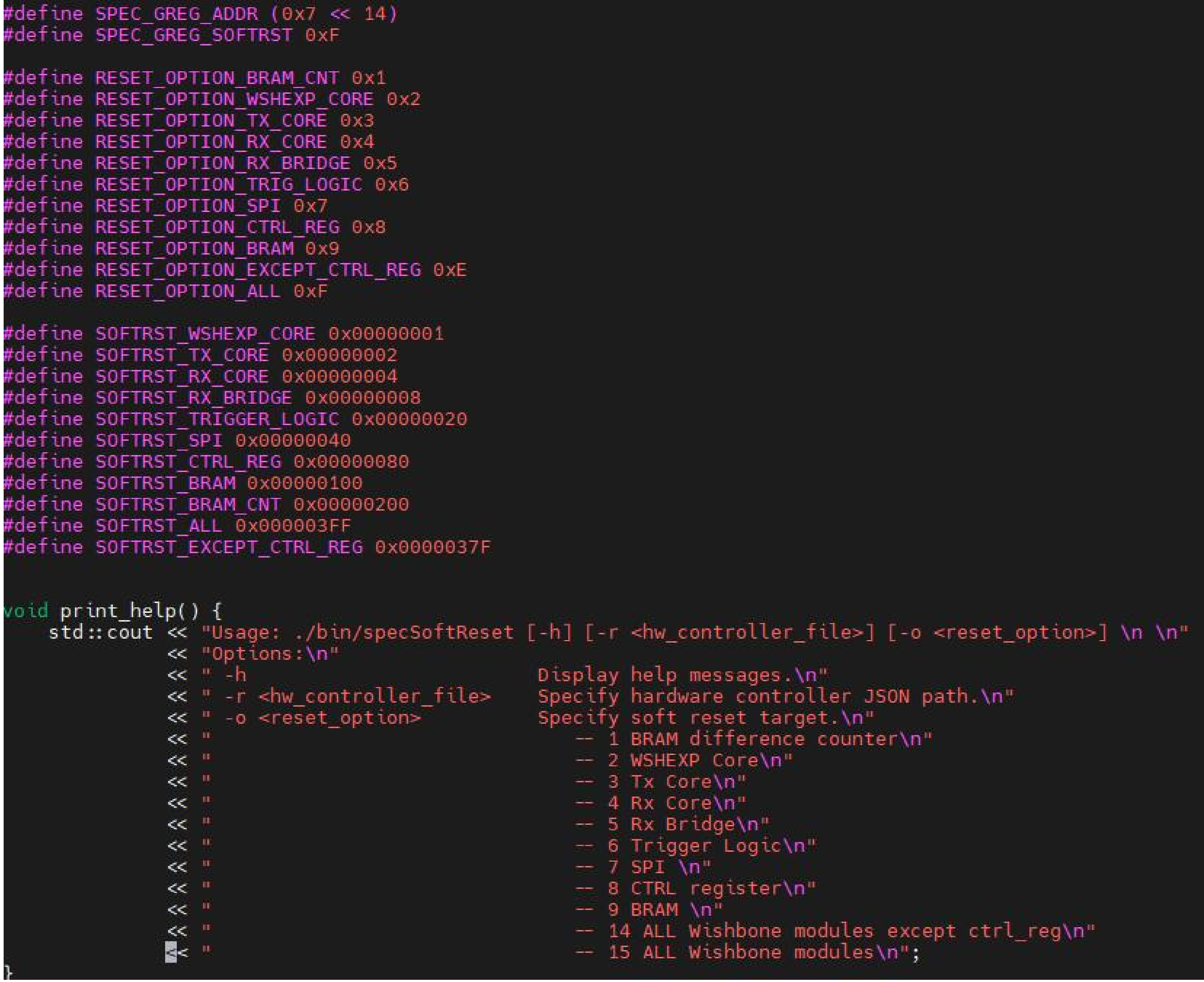
- After running some YARR software side scan/app/test, the FW can stuck at some weird state. To perform another scan/app/test, we need to reset the FW but the only way was to reboot the host.
- For busy logic in BRAM, some scan/app/test can cause the busy counter to be misaligned, we need to reset the counter before doing a scan.

How is soft reset achieved on FW

- In control register module, when it receive a write to soft reset register, it set target's soft reset signal high for 8 cycles.
- These soft reset signal then goes into corresponding modules and reset internal signals just like rebooting host.

How to do soft reset with executable

- After building YARR SW, there will be a executable under ./bin called specSoftReset
- -h option will show the help
- -r option is used to get the correct specCom Id
- -o option let user choose which module to reset

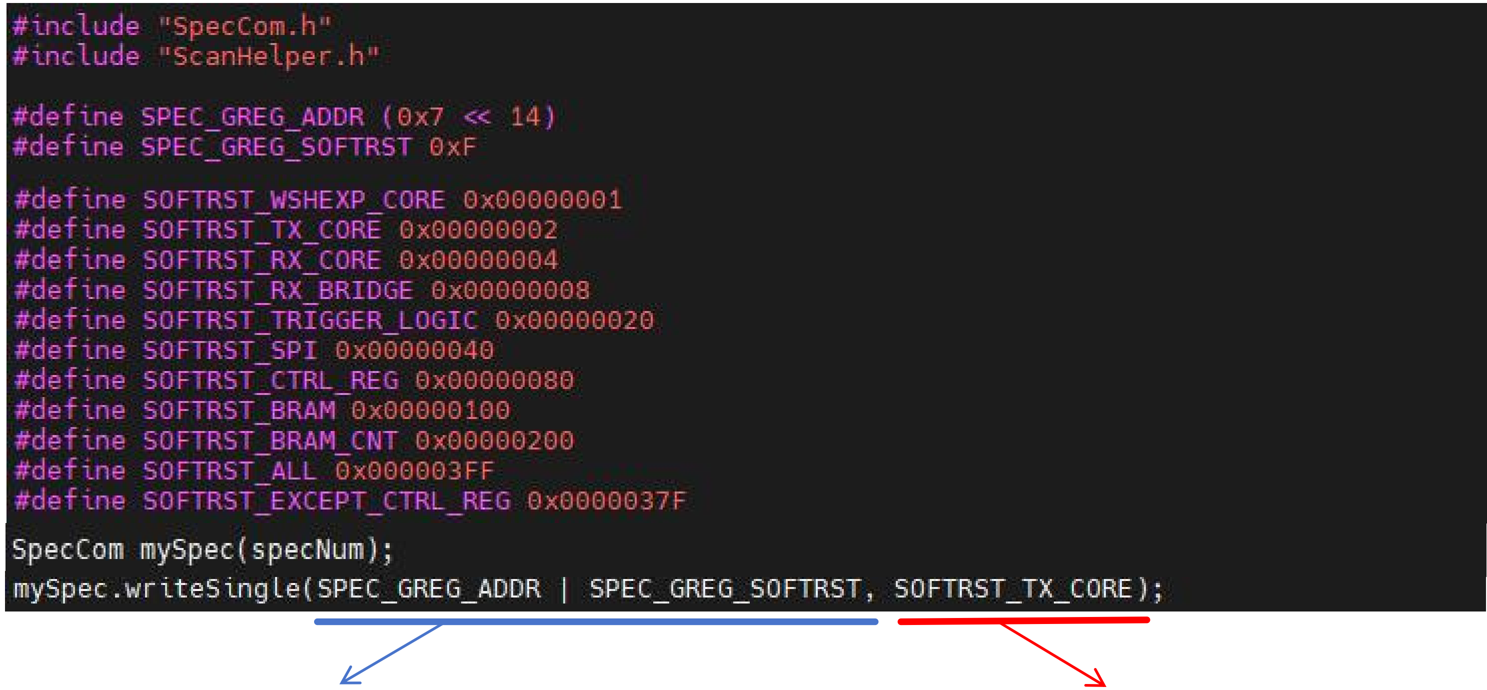


```
#define SPEC_GREG_ADDR (0x7 << 14)
#define SPEC_GREG_SOFTRST 0xF

#define RESET_OPTION_BRAM_CNT 0x1
#define RESET_OPTION_WSHEXP_CORE 0x2
#define RESET_OPTION_TX_CORE 0x3
#define RESET_OPTION_RX_CORE 0x4
#define RESET_OPTION_RX_BRIDGE 0x5
#define RESET_OPTION_TRIG_LOGIC 0x6
#define RESET_OPTION_SPI 0x7
#define RESET_OPTION_CTRL_REG 0x8
#define RESET_OPTION_BRAM 0x9
#define RESET_OPTION_EXCEPT_CTRL_REG 0xE
#define RESET_OPTION_ALL 0xF

#define SOFTRST_WSHEXP_CORE 0x00000001
#define SOFTRST_TX_CORE 0x00000002
#define SOFTRST_RX_CORE 0x00000004
#define SOFTRST_RX_BRIDGE 0x00000008
#define SOFTRST_TRIGGER_LOGIC 0x00000020
#define SOFTRST_SPI 0x00000040
#define SOFTRST_CTRL_REG 0x00000080
#define SOFTRST_BRAM 0x00000100
#define SOFTRST_BRAM_CNT 0x00000200
#define SOFTRST_ALL 0x000003FF
#define SOFTRST_EXCEPT_CTRL_REG 0x0000037F


void print_help() {
    std::cout << "Usage: ./bin/specSoftReset [-h] [-r <hw_controller_file>] [-o <reset_option>] \n \n"
              << "Options:\n"
              << " -h                             Display help messages.\n"
              << " -r <hw_controller_file>        Specify hardware controller JSON path.\n"
              << " -o <reset_option>              Specify soft reset target.\n"
              << "                                   -- 1 BRAM difference counter\n"
              << "                                   -- 2 WSHEXP Core\n"
              << "                                   -- 3 Tx Core\n"
              << "                                   -- 4 Rx Core\n"
              << "                                   -- 5 Rx Bridge\n"
              << "                                   -- 6 Trigger Logic\n"
              << "                                   -- 7 SPI \n"
              << "                                   -- 8 CTRL register\n"
              << "                                   -- 9 BRAM \n"
              << "                                   -- 14 ALL Wishbone modules except ctrl_reg\n"
              << "                                   -- 15 ALL Wishbone modules\n";
}
```

Example: ./bin/specSoftReset -r configs/controller/specCfg-rd53b-16x1.json -o 14

How to do soft reset with in line code

- To add soft reset in your own software side application, you need to include “specCom.h” and find correct specNum
- Then, use writeSingle function in SpecCom you can write the data with SoftResetTarget into the SoftResetRegister on FW
- For specNum, you can find it in controller config file with ScanHelper.h



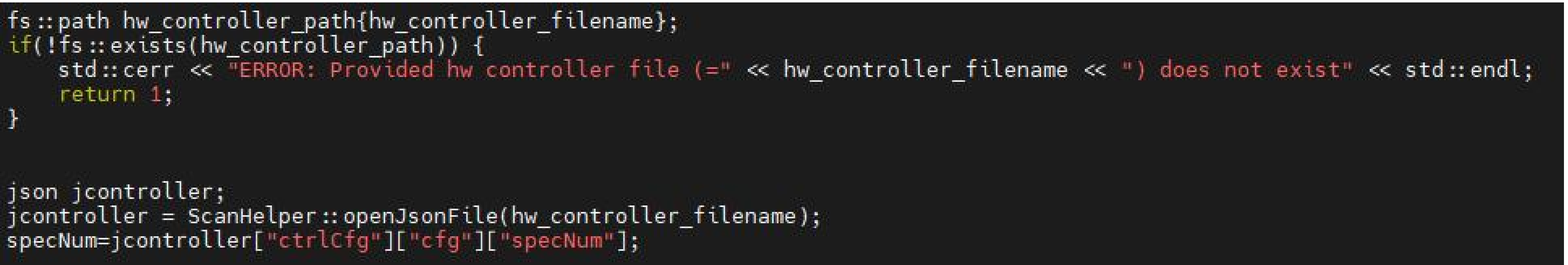
```
#include "SpecCom.h"
#include "ScanHelper.h"

#define SPEC_GREG_ADDR (0x7 << 14)
#define SPEC_GREG_SOFTRST 0xF

#define SOFTRST_WSHEXP_CORE 0x00000001
#define SOFTRST_TX_CORE 0x00000002
#define SOFTRST_RX_CORE 0x00000004
#define SOFTRST_RX_BRIDGE 0x00000008
#define SOFTRST_TRIGGER_LOGIC 0x00000020
#define SOFTRST_SPI 0x00000040
#define SOFTRST_CTRL_REG 0x00000080
#define SOFTRST_BRAM 0x00000100
#define SOFTRST_BRAM_CNT 0x00000200
#define SOFTRST_ALL 0x000003FF
#define SOFTRST_EXCEPT_CTRL_REG 0x0000037F
SpecCom mySpec(specNum);
mySpec.writeSingle(SPEC_GREG_ADDR | SPEC_GREG_SOFTRST, SOFTRST_TX_CORE);
```

Soft Reset Register Addr

Soft Reset Target

Each bit in write data represents a reset signal for modules in FW.

If you want to reset multiple modules, you can ‘OR’ the definitions to create your own target list.
E.g. SOFTRST_TX_CORE | SOFTRST_RX_CORE
will reset both wb_Tx_core and wb_Rx_core



```
fs::path hw_controller_path{hw_controller_filename};
if(!fs::exists(hw_controller_path)) {
    std::cerr << "ERROR: Provided hw controller file (=" << hw_controller_filename << ") does not exist" << std::endl;
    return 1;
}


json jcontroller;
jcontroller = ScanHelper::openJsonFile(hw_controller_filename);
specNum=jcontroller["ctrlCfg"]["cfg"]["specNum"];
```

How add new soft reset

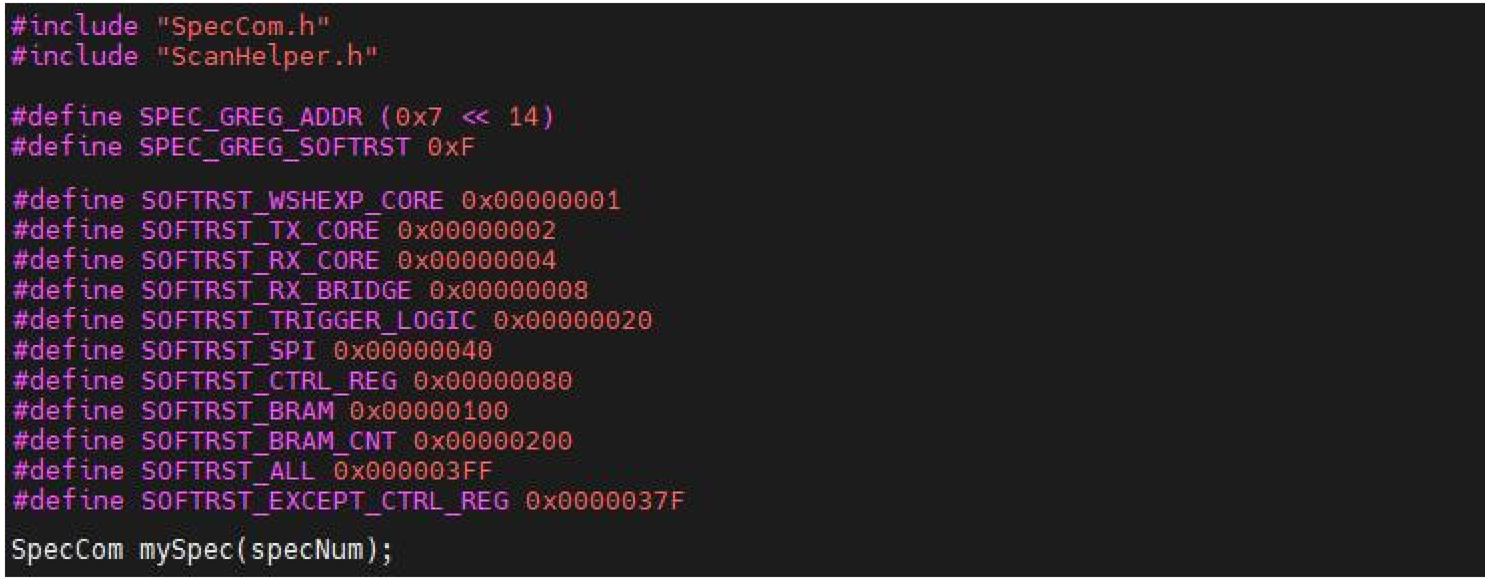
- Currently, we only used 10 bits in the soft reset register. User can add their own soft reset in the future.

FW side

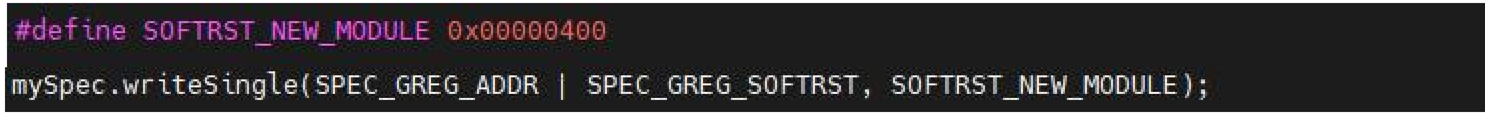
Example:

```
   new_module
   port map{
      ...
      rst_i => not rst_n_s,
      ...
   }
 new_module
 port map{
    ...
    rst_i => (not rst_n_s) or soft_rst_reg(10),
    ...
 }
```

SW side



```
#include "SpecCom.h"
#include "ScanHelper.h"

#define SPEC_GREG_ADDR (0x7 << 14)
#define SPEC_GREG_SOFTRST 0xF

#define SOFTRST_WSHEXP_CORE 0x00000001
#define SOFTRST_TX_CORE 0x00000002
#define SOFTRST_RX_CORE 0x00000004
#define SOFTRST_RX_BRIDGE 0x00000008
#define SOFTRST_TRIGGER_LOGIC 0x00000020
#define SOFTRST_SPI 0x00000040
#define SOFTRST_CTRL_REG 0x00000080
#define SOFTRST_BRAM 0x00000100
#define SOFTRST_BRAM_CNT 0x00000200
#define SOFTRST_ALL 0x000003FF
#define SOFTRST_EXCEPT_CTRL_REG 0x0000037F

SpecCom mySpec(specNum);
```

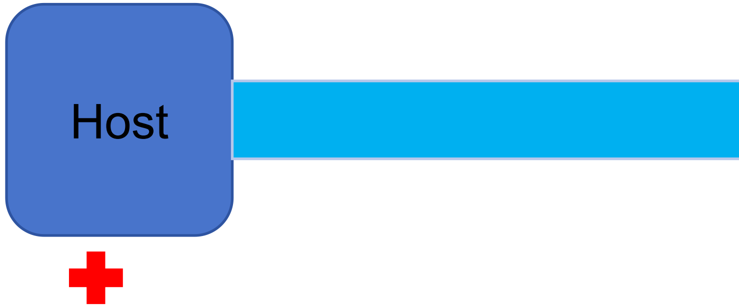


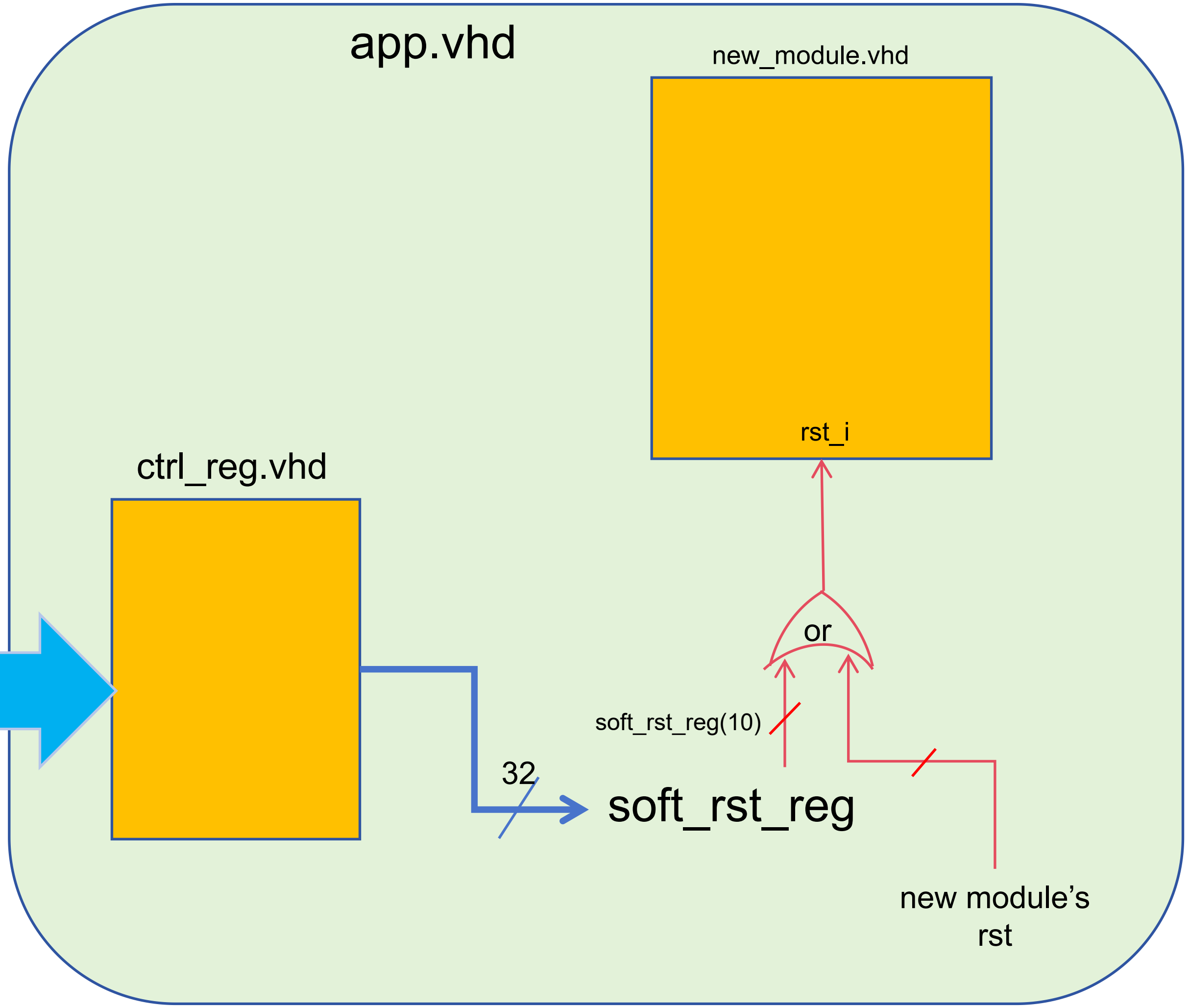
```
#define SOFTRST_NEW_MODULE 0x00000400

mySpec.writeSingle(SPEC_GREG_ADDR | SPEC_GREG_SOFTRST, SOFTRST_NEW_MODULE);
```

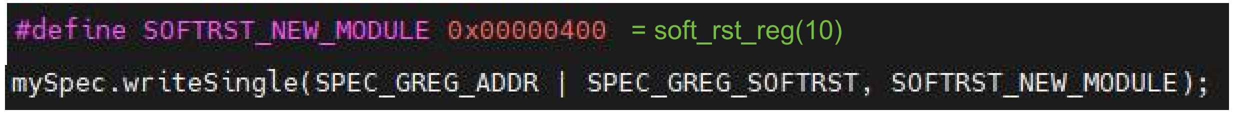
How add new soft reset

What you need to add:

- SW side
  - definition of bit in register representing the module.
- FW side
  - logics marked red.







```
#define SOFTRST_NEW_MODULE 0x00000400  = soft_rst_reg(10)
mySpec.writeSingle(SPEC_GREG_ADDR | SPEC_GREG_SOFTRST, SOFTRST_NEW_MODULE);
```