// ==============================================================

// RTL generated by Vivado(TM) HLS - High-Level Synthesis from C, C++ and SystemC

// Version: 2017.2

// Copyright (C) 1986-2017 Xilinx, Inc. All Rights Reserved.

//

// ===========================================================

#ifndef \_matrixmul\_HH\_

#define \_matrixmul\_HH\_

#include "systemc.h"

#include "AESL\_pkg.h"

#include "matrixmul\_mac\_mulbkb.h"

#include "matrixmul\_mac\_mulcud.h"

namespace ap\_rtl {

struct matrixmul : public sc\_module {

// Port declarations 41

sc\_in\_clk ap\_clk;

sc\_in< sc\_logic > ap\_rst;

sc\_in< sc\_logic > ap\_start;

sc\_out< sc\_logic > ap\_done;

sc\_out< sc\_logic > ap\_idle;

sc\_out< sc\_logic > ap\_ready;

sc\_out< sc\_lv<2> > a\_0\_address0;

sc\_out< sc\_logic > a\_0\_ce0;

sc\_in< sc\_lv<8> > a\_0\_q0;

sc\_out< sc\_lv<2> > a\_1\_address0;

sc\_out< sc\_logic > a\_1\_ce0;

sc\_in< sc\_lv<8> > a\_1\_q0;

sc\_out< sc\_lv<2> > a\_2\_address0;

sc\_out< sc\_logic > a\_2\_ce0;

sc\_in< sc\_lv<8> > a\_2\_q0;

sc\_out< sc\_lv<2> > b\_0\_address0;

sc\_out< sc\_logic > b\_0\_ce0;

sc\_in< sc\_lv<8> > b\_0\_q0;

sc\_out< sc\_lv<2> > b\_0\_address1;

sc\_out< sc\_logic > b\_0\_ce1;

sc\_in< sc\_lv<8> > b\_0\_q1;

sc\_out< sc\_lv<2> > b\_1\_address0;

sc\_out< sc\_logic > b\_1\_ce0;

sc\_in< sc\_lv<8> > b\_1\_q0;

sc\_out< sc\_lv<2> > b\_1\_address1;

sc\_out< sc\_logic > b\_1\_ce1;

sc\_in< sc\_lv<8> > b\_1\_q1;

sc\_out< sc\_lv<2> > b\_2\_address0;

sc\_out< sc\_logic > b\_2\_ce0;

sc\_in< sc\_lv<8> > b\_2\_q0;

sc\_out< sc\_lv<2> > b\_2\_address1;

sc\_out< sc\_logic > b\_2\_ce1;

sc\_in< sc\_lv<8> > b\_2\_q1;

sc\_out< sc\_lv<4> > res\_address0;

sc\_out< sc\_logic > res\_ce0;

sc\_out< sc\_logic > res\_we0;

sc\_out< sc\_lv<16> > res\_d0;

sc\_out< sc\_lv<4> > res\_address1;

sc\_out< sc\_logic > res\_ce1;

sc\_out< sc\_logic > res\_we1;

sc\_out< sc\_lv<16> > res\_d1;

// Module declarations

matrixmul(sc\_module\_name name);

SC\_HAS\_PROCESS(matrixmul);

~matrixmul();

sc\_trace\_file\* mVcdFile;

ofstream mHdltvinHandle;

ofstream mHdltvoutHandle;

matrixmul\_mac\_mulbkb<1,1,8,8,16,16>\* matrixmul\_mac\_mulbkb\_U1;

matrixmul\_mac\_mulbkb<1,1,8,8,16,16>\* matrixmul\_mac\_mulbkb\_U2;

matrixmul\_mac\_mulbkb<1,1,8,8,16,16>\* matrixmul\_mac\_mulbkb\_U3;

matrixmul\_mac\_mulcud<1,1,8,8,16,16>\* matrixmul\_mac\_mulcud\_U4;

matrixmul\_mac\_mulcud<1,1,8,8,16,16>\* matrixmul\_mac\_mulcud\_U5;

matrixmul\_mac\_mulcud<1,1,8,8,16,16>\* matrixmul\_mac\_mulcud\_U6;

sc\_signal< sc\_lv<4> > ap\_CS\_fsm;

sc\_signal< sc\_logic > ap\_CS\_fsm\_state1;

sc\_signal< sc\_lv<2> > i\_reg\_220;

sc\_signal< sc\_lv<2> > ap\_reg\_pp0\_iter1\_i\_reg\_220;

sc\_signal< sc\_logic > ap\_CS\_fsm\_pp0\_stage0;

sc\_signal< bool > ap\_block\_state2\_pp0\_stage0\_iter0;

sc\_signal< bool > ap\_block\_state4\_pp0\_stage0\_iter1;

sc\_signal< bool > ap\_block\_state6\_pp0\_stage0\_iter2;

sc\_signal< bool > ap\_block\_pp0\_stage0\_flag00011001;

sc\_signal< sc\_lv<8> > reg\_232;

sc\_signal< sc\_logic > ap\_CS\_fsm\_pp0\_stage1;

sc\_signal< sc\_logic > ap\_enable\_reg\_pp0\_iter0;

sc\_signal< bool > ap\_block\_state3\_pp0\_stage1\_iter0;

sc\_signal< bool > ap\_block\_state5\_pp0\_stage1\_iter1;

sc\_signal< bool > ap\_block\_pp0\_stage1\_flag00011001;

sc\_signal< sc\_lv<1> > exitcond2\_reg\_463;

sc\_signal< sc\_logic > ap\_enable\_reg\_pp0\_iter1;

sc\_signal< sc\_lv<8> > reg\_236;

sc\_signal< sc\_lv<8> > reg\_240;

sc\_signal< sc\_lv<1> > exitcond2\_fu\_244\_p2;

sc\_signal< sc\_lv<1> > ap\_reg\_pp0\_iter1\_exitcond2\_reg\_463;

sc\_signal< sc\_lv<2> > i\_1\_fu\_250\_p2;

sc\_signal< sc\_lv<2> > i\_1\_reg\_467;

sc\_signal< sc\_lv<8> > a\_0\_load\_reg\_487;

sc\_signal< sc\_lv<8> > a\_1\_load\_reg\_492;

sc\_signal< sc\_lv<16> > tmp\_5\_0\_2\_fu\_263\_p1;

sc\_signal< sc\_lv<16> > tmp\_5\_0\_2\_reg\_497;

sc\_signal< sc\_lv<16> > tmp\_7\_0\_2\_fu\_271\_p2;

sc\_signal< sc\_lv<16> > tmp\_7\_0\_2\_reg\_502;

sc\_signal< sc\_lv<16> > tmp\_7\_1\_2\_fu\_281\_p2;

sc\_signal< sc\_lv<16> > tmp\_7\_1\_2\_reg\_507;

sc\_signal< sc\_lv<8> > b\_2\_load\_2\_reg\_512;

sc\_signal< sc\_lv<16> > grp\_fu\_373\_p3;

sc\_signal< sc\_lv<16> > tmp1\_reg\_517;

sc\_signal< sc\_lv<16> > grp\_fu\_380\_p3;

sc\_signal< sc\_lv<16> > tmp2\_reg\_522;

sc\_signal< sc\_lv<16> > grp\_fu\_387\_p3;

sc\_signal< sc\_lv<16> > tmp3\_reg\_527;

sc\_signal< sc\_lv<5> > tmp\_3\_fu\_326\_p2;

sc\_signal< sc\_lv<5> > tmp\_3\_reg\_532;

sc\_signal< sc\_lv<16> > grp\_fu\_411\_p3;

sc\_signal< sc\_lv<16> > tmp\_8\_2\_2\_reg\_537;

sc\_signal< bool > ap\_block\_pp0\_stage0\_flag00011011;

sc\_signal< sc\_logic > ap\_condition\_pp0\_exit\_iter0\_state2;

sc\_signal< bool > ap\_block\_pp0\_stage1\_flag00011011;

sc\_signal< sc\_logic > ap\_enable\_reg\_pp0\_iter2;

sc\_signal< sc\_lv<2> > i\_phi\_fu\_224\_p4;

sc\_signal< bool > ap\_block\_pp0\_stage0\_flag00000000;

sc\_signal< sc\_lv<64> > tmp\_fu\_256\_p1;

sc\_signal< sc\_lv<64> > tmp\_3\_cast\_fu\_332\_p1;

sc\_signal< bool > ap\_block\_pp0\_stage1\_flag00000000;

sc\_signal< sc\_lv<64> > tmp\_4\_cast\_fu\_343\_p1;

sc\_signal< sc\_lv<64> > tmp\_8\_cast\_fu\_368\_p1;

sc\_signal< sc\_lv<16> > grp\_fu\_395\_p3;

sc\_signal< sc\_lv<16> > grp\_fu\_403\_p3;

sc\_signal< sc\_lv<8> > tmp\_7\_0\_2\_fu\_271\_p0;

sc\_signal< sc\_lv<8> > tmp\_7\_0\_2\_fu\_271\_p1;

sc\_signal< sc\_lv<8> > tmp\_7\_1\_2\_fu\_281\_p0;

sc\_signal< sc\_lv<8> > tmp\_7\_1\_2\_fu\_281\_p1;

sc\_signal< sc\_lv<8> > tmp\_7\_2\_2\_fu\_305\_p0;

sc\_signal< sc\_lv<8> > tmp\_7\_2\_2\_fu\_305\_p1;

sc\_signal< sc\_lv<4> > tmp\_2\_fu\_314\_p3;

sc\_signal< sc\_lv<5> > p\_shl\_cast\_fu\_322\_p1;

sc\_signal< sc\_lv<5> > tmp\_cast\_fu\_310\_p1;

sc\_signal< sc\_lv<5> > tmp\_4\_fu\_337\_p2;

sc\_signal< sc\_lv<5> > tmp\_8\_fu\_363\_p2;

sc\_signal< sc\_lv<8> > grp\_fu\_373\_p1;

sc\_signal< sc\_lv<16> > tmp\_5\_fu\_287\_p1;

sc\_signal< sc\_lv<8> > grp\_fu\_380\_p1;

sc\_signal< sc\_lv<8> > grp\_fu\_387\_p1;

sc\_signal< sc\_lv<16> > grp\_fu\_387\_p2;

sc\_signal< sc\_lv<8> > grp\_fu\_395\_p1;

sc\_signal< sc\_lv<16> > tmp\_5\_0\_1\_fu\_348\_p1;

sc\_signal< sc\_lv<8> > grp\_fu\_403\_p1;

sc\_signal< sc\_lv<8> > grp\_fu\_411\_p1;

sc\_signal< sc\_logic > ap\_CS\_fsm\_state7;

sc\_signal< sc\_lv<4> > ap\_NS\_fsm;

sc\_signal< sc\_logic > ap\_idle\_pp0;

sc\_signal< sc\_logic > ap\_enable\_pp0;

static const sc\_logic ap\_const\_logic\_1;

static const sc\_logic ap\_const\_logic\_0;

static const sc\_lv<4> ap\_ST\_fsm\_state1;

static const sc\_lv<4> ap\_ST\_fsm\_pp0\_stage0;

static const sc\_lv<4> ap\_ST\_fsm\_pp0\_stage1;

static const sc\_lv<4> ap\_ST\_fsm\_state7;

static const sc\_lv<32> ap\_const\_lv32\_0;

static const bool ap\_const\_boolean\_1;

static const sc\_lv<32> ap\_const\_lv32\_1;

static const bool ap\_const\_boolean\_0;

static const sc\_lv<32> ap\_const\_lv32\_2;

static const sc\_lv<1> ap\_const\_lv1\_0;

static const sc\_lv<1> ap\_const\_lv1\_1;

static const sc\_lv<2> ap\_const\_lv2\_0;

static const sc\_lv<64> ap\_const\_lv64\_0;

static const sc\_lv<64> ap\_const\_lv64\_1;

static const sc\_lv<64> ap\_const\_lv64\_2;

static const sc\_lv<2> ap\_const\_lv2\_3;

static const sc\_lv<2> ap\_const\_lv2\_1;

static const sc\_lv<5> ap\_const\_lv5\_1;

static const sc\_lv<5> ap\_const\_lv5\_2;

static const sc\_lv<32> ap\_const\_lv32\_3;

// Thread declarations

void thread\_ap\_clk\_no\_reset\_();

void thread\_a\_0\_address0();

void thread\_a\_0\_ce0();

void thread\_a\_1\_address0();

void thread\_a\_1\_ce0();

void thread\_a\_2\_address0();

void thread\_a\_2\_ce0();

void thread\_ap\_CS\_fsm\_pp0\_stage0();

void thread\_ap\_CS\_fsm\_pp0\_stage1();

void thread\_ap\_CS\_fsm\_state1();

void thread\_ap\_CS\_fsm\_state7();

void thread\_ap\_block\_pp0\_stage0\_flag00000000();

void thread\_ap\_block\_pp0\_stage0\_flag00011001();

void thread\_ap\_block\_pp0\_stage0\_flag00011011();

void thread\_ap\_block\_pp0\_stage1\_flag00000000();

void thread\_ap\_block\_pp0\_stage1\_flag00011001();

void thread\_ap\_block\_pp0\_stage1\_flag00011011();

void thread\_ap\_block\_state2\_pp0\_stage0\_iter0();

void thread\_ap\_block\_state3\_pp0\_stage1\_iter0();

void thread\_ap\_block\_state4\_pp0\_stage0\_iter1();

void thread\_ap\_block\_state5\_pp0\_stage1\_iter1();

void thread\_ap\_block\_state6\_pp0\_stage0\_iter2();

void thread\_ap\_condition\_pp0\_exit\_iter0\_state2();

void thread\_ap\_done();

void thread\_ap\_enable\_pp0();

void thread\_ap\_idle();

void thread\_ap\_idle\_pp0();

void thread\_ap\_ready();

void thread\_b\_0\_address0();

void thread\_b\_0\_address1();

void thread\_b\_0\_ce0();

void thread\_b\_0\_ce1();

void thread\_b\_1\_address0();

void thread\_b\_1\_address1();

void thread\_b\_1\_ce0();

void thread\_b\_1\_ce1();

void thread\_b\_2\_address0();

void thread\_b\_2\_address1();

void thread\_b\_2\_ce0();

void thread\_b\_2\_ce1();

void thread\_exitcond2\_fu\_244\_p2();

void thread\_grp\_fu\_373\_p1();

void thread\_grp\_fu\_380\_p1();

void thread\_grp\_fu\_387\_p1();

void thread\_grp\_fu\_387\_p2();

void thread\_grp\_fu\_395\_p1();

void thread\_grp\_fu\_403\_p1();

void thread\_grp\_fu\_411\_p1();

void thread\_i\_1\_fu\_250\_p2();

void thread\_i\_phi\_fu\_224\_p4();

void thread\_p\_shl\_cast\_fu\_322\_p1();

void thread\_res\_address0();

void thread\_res\_address1();

void thread\_res\_ce0();

void thread\_res\_ce1();

void thread\_res\_d0();

void thread\_res\_d1();

void thread\_res\_we0();

void thread\_res\_we1();

void thread\_tmp\_2\_fu\_314\_p3();

void thread\_tmp\_3\_cast\_fu\_332\_p1();

void thread\_tmp\_3\_fu\_326\_p2();

void thread\_tmp\_4\_cast\_fu\_343\_p1();

void thread\_tmp\_4\_fu\_337\_p2();

void thread\_tmp\_5\_0\_1\_fu\_348\_p1();

void thread\_tmp\_5\_0\_2\_fu\_263\_p1();

void thread\_tmp\_5\_fu\_287\_p1();

void thread\_tmp\_7\_0\_2\_fu\_271\_p0();

void thread\_tmp\_7\_0\_2\_fu\_271\_p1();

void thread\_tmp\_7\_0\_2\_fu\_271\_p2();

void thread\_tmp\_7\_1\_2\_fu\_281\_p0();

void thread\_tmp\_7\_1\_2\_fu\_281\_p1();

void thread\_tmp\_7\_1\_2\_fu\_281\_p2();

void thread\_tmp\_7\_2\_2\_fu\_305\_p0();

void thread\_tmp\_7\_2\_2\_fu\_305\_p1();

void thread\_tmp\_8\_cast\_fu\_368\_p1();

void thread\_tmp\_8\_fu\_363\_p2();

void thread\_tmp\_cast\_fu\_310\_p1();

void thread\_tmp\_fu\_256\_p1();

void thread\_ap\_NS\_fsm();

void thread\_hdltv\_gen();

};

}

using namespace ap\_rtl;

#endif