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

// 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 \_simple\_HH\_

#define \_simple\_HH\_

#include "systemc.h"

#include "AESL\_pkg.h"

#include "simple\_fadd\_32ns\_bkb.h"

namespace ap\_rtl {

struct simple : public sc\_module {

// Port declarations 17

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<3> > a\_address0;

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

sc\_in< sc\_lv<32> > a\_q0;

sc\_out< sc\_lv<3> > b\_address0;

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

sc\_in< sc\_lv<32> > b\_q0;

sc\_out< sc\_lv<3> > c\_address0;

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

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

sc\_out< sc\_lv<32> > c\_d0;

sc\_out< sc\_lv<32> > ap\_return;

sc\_signal< sc\_logic > ap\_var\_for\_const0;

// Module declarations

simple(sc\_module\_name name);

SC\_HAS\_PROCESS(simple);

~simple();

sc\_trace\_file\* mVcdFile;

ofstream mHdltvinHandle;

ofstream mHdltvoutHandle;

simple\_fadd\_32ns\_bkb<1,5,32,32,32>\* simple\_fadd\_32ns\_bkb\_U1;

sc\_signal< sc\_lv<7> > ap\_CS\_fsm;

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

sc\_signal< sc\_lv<4> > i\_1\_fu\_86\_p2;

sc\_signal< sc\_lv<4> > i\_1\_reg\_101;

sc\_signal< sc\_logic > ap\_CS\_fsm\_state2;

sc\_signal< sc\_lv<64> > tmp\_fu\_92\_p1;

sc\_signal< sc\_lv<64> > tmp\_reg\_106;

sc\_signal< sc\_lv<1> > exitcond\_fu\_80\_p2;

sc\_signal< sc\_logic > ap\_CS\_fsm\_state3;

sc\_signal< sc\_lv<4> > i\_reg\_62;

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

sc\_signal< sc\_lv<32> > grp\_fu\_73\_p2;

sc\_signal< sc\_lv<7> > ap\_NS\_fsm;

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

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

static const sc\_lv<7> ap\_ST\_fsm\_state1;

static const sc\_lv<7> ap\_ST\_fsm\_state2;

static const sc\_lv<7> ap\_ST\_fsm\_state3;

static const sc\_lv<7> ap\_ST\_fsm\_state4;

static const sc\_lv<7> ap\_ST\_fsm\_state5;

static const sc\_lv<7> ap\_ST\_fsm\_state6;

static const sc\_lv<7> ap\_ST\_fsm\_state7;

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

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

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

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

static const sc\_lv<4> ap\_const\_lv4\_0;

static const sc\_lv<32> ap\_const\_lv32\_6;

static const sc\_lv<4> ap\_const\_lv4\_8;

static const sc\_lv<4> ap\_const\_lv4\_1;

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

static const bool ap\_const\_boolean\_1;

// Thread declarations

void thread\_ap\_var\_for\_const0();

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

void thread\_a\_address0();

void thread\_a\_ce0();

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

void thread\_ap\_CS\_fsm\_state2();

void thread\_ap\_CS\_fsm\_state3();

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

void thread\_ap\_done();

void thread\_ap\_idle();

void thread\_ap\_ready();

void thread\_ap\_return();

void thread\_b\_address0();

void thread\_b\_ce0();

void thread\_c\_address0();

void thread\_c\_ce0();

void thread\_c\_d0();

void thread\_c\_we0();

void thread\_exitcond\_fu\_80\_p2();

void thread\_i\_1\_fu\_86\_p2();

void thread\_tmp\_fu\_92\_p1();

void thread\_ap\_NS\_fsm();

void thread\_hdltv\_gen();

};

}

using namespace ap\_rtl;

#endif