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

// 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.

//

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

#include "matrixmul.h"

#include "AESL\_pkg.h"

using namespace std;

namespace ap\_rtl {

const sc\_logic matrixmul::ap\_const\_logic\_1 = sc\_dt::Log\_1;

const sc\_logic matrixmul::ap\_const\_logic\_0 = sc\_dt::Log\_0;

const sc\_lv<5> matrixmul::ap\_ST\_fsm\_state1 = "1";

const sc\_lv<5> matrixmul::ap\_ST\_fsm\_state2 = "10";

const sc\_lv<5> matrixmul::ap\_ST\_fsm\_state3 = "100";

const sc\_lv<5> matrixmul::ap\_ST\_fsm\_state4 = "1000";

const sc\_lv<5> matrixmul::ap\_ST\_fsm\_state5 = "10000";

const sc\_lv<32> matrixmul::ap\_const\_lv32\_0 = "00000000000000000000000000000000";

const sc\_lv<32> matrixmul::ap\_const\_lv32\_1 = "1";

const sc\_lv<1> matrixmul::ap\_const\_lv1\_0 = "0";

const sc\_lv<32> matrixmul::ap\_const\_lv32\_2 = "10";

const sc\_lv<32> matrixmul::ap\_const\_lv32\_3 = "11";

const sc\_lv<32> matrixmul::ap\_const\_lv32\_4 = "100";

const sc\_lv<2> matrixmul::ap\_const\_lv2\_0 = "00";

const sc\_lv<1> matrixmul::ap\_const\_lv1\_1 = "1";

const sc\_lv<16> matrixmul::ap\_const\_lv16\_0 = "0000000000000000";

const sc\_lv<2> matrixmul::ap\_const\_lv2\_3 = "11";

const sc\_lv<2> matrixmul::ap\_const\_lv2\_1 = "1";

const bool matrixmul::ap\_const\_boolean\_1 = true;

matrixmul::matrixmul(sc\_module\_name name) : sc\_module(name), mVcdFile(0) {

matrixmul\_mac\_mulbkb\_U1 = new matrixmul\_mac\_mulbkb<1,1,8,8,16,16>("matrixmul\_mac\_mulbkb\_U1");

matrixmul\_mac\_mulbkb\_U1->din0(b\_q0);

matrixmul\_mac\_mulbkb\_U1->din1(a\_q0);

matrixmul\_mac\_mulbkb\_U1->din2(res\_load\_reg\_97);

matrixmul\_mac\_mulbkb\_U1->dout(grp\_fu\_243\_p3);

SC\_METHOD(thread\_ap\_clk\_no\_reset\_);

dont\_initialize();

sensitive << ( ap\_clk.pos() );

SC\_METHOD(thread\_a\_address0);

sensitive << ( ap\_CS\_fsm\_state4 );

sensitive << ( tmp\_12\_cast\_fu\_202\_p1 );

SC\_METHOD(thread\_a\_ce0);

sensitive << ( ap\_CS\_fsm\_state4 );

SC\_METHOD(thread\_ap\_CS\_fsm\_state1);

sensitive << ( ap\_CS\_fsm );

SC\_METHOD(thread\_ap\_CS\_fsm\_state2);

sensitive << ( ap\_CS\_fsm );

SC\_METHOD(thread\_ap\_CS\_fsm\_state3);

sensitive << ( ap\_CS\_fsm );

SC\_METHOD(thread\_ap\_CS\_fsm\_state4);

sensitive << ( ap\_CS\_fsm );

SC\_METHOD(thread\_ap\_CS\_fsm\_state5);

sensitive << ( ap\_CS\_fsm );

SC\_METHOD(thread\_ap\_done);

sensitive << ( ap\_CS\_fsm\_state2 );

sensitive << ( exitcond2\_fu\_121\_p2 );

SC\_METHOD(thread\_ap\_idle);

sensitive << ( ap\_start );

sensitive << ( ap\_CS\_fsm\_state1 );

SC\_METHOD(thread\_ap\_ready);

sensitive << ( ap\_CS\_fsm\_state2 );

sensitive << ( exitcond2\_fu\_121\_p2 );

SC\_METHOD(thread\_b\_address0);

sensitive << ( ap\_CS\_fsm\_state4 );

sensitive << ( tmp\_15\_cast\_fu\_230\_p1 );

SC\_METHOD(thread\_b\_ce0);

sensitive << ( ap\_CS\_fsm\_state4 );

SC\_METHOD(thread\_exitcond1\_fu\_155\_p2);

sensitive << ( ap\_CS\_fsm\_state3 );

sensitive << ( j\_reg\_86 );

SC\_METHOD(thread\_exitcond2\_fu\_121\_p2);

sensitive << ( ap\_CS\_fsm\_state2 );

sensitive << ( i\_reg\_75 );

SC\_METHOD(thread\_exitcond\_fu\_181\_p2);

sensitive << ( ap\_CS\_fsm\_state4 );

sensitive << ( k\_reg\_110 );

SC\_METHOD(thread\_i\_1\_fu\_127\_p2);

sensitive << ( i\_reg\_75 );

SC\_METHOD(thread\_j\_1\_fu\_161\_p2);

sensitive << ( j\_reg\_86 );

SC\_METHOD(thread\_k\_1\_fu\_187\_p2);

sensitive << ( k\_reg\_110 );

SC\_METHOD(thread\_p\_shl1\_cast\_fu\_215\_p1);

sensitive << ( tmp\_10\_fu\_207\_p3 );

SC\_METHOD(thread\_p\_shl\_cast\_fu\_145\_p1);

sensitive << ( tmp\_9\_fu\_137\_p3 );

SC\_METHOD(thread\_res\_address0);

sensitive << ( res\_addr\_reg\_278 );

sensitive << ( ap\_CS\_fsm\_state4 );

SC\_METHOD(thread\_res\_ce0);

sensitive << ( ap\_CS\_fsm\_state4 );

SC\_METHOD(thread\_res\_d0);

sensitive << ( ap\_CS\_fsm\_state4 );

sensitive << ( res\_load\_reg\_97 );

SC\_METHOD(thread\_res\_we0);

sensitive << ( ap\_CS\_fsm\_state4 );

SC\_METHOD(thread\_tmp\_10\_fu\_207\_p3);

sensitive << ( k\_reg\_110 );

SC\_METHOD(thread\_tmp\_11\_cast\_fu\_176\_p1);

sensitive << ( tmp\_2\_fu\_171\_p2 );

SC\_METHOD(thread\_tmp\_11\_fu\_219\_p2);

sensitive << ( tmp\_4\_cast\_fu\_193\_p1 );

sensitive << ( p\_shl1\_cast\_fu\_215\_p1 );

SC\_METHOD(thread\_tmp\_12\_cast\_fu\_202\_p1);

sensitive << ( tmp\_4\_fu\_197\_p2 );

SC\_METHOD(thread\_tmp\_12\_fu\_225\_p2);

sensitive << ( tmp\_2\_cast\_reg\_273 );

sensitive << ( tmp\_11\_fu\_219\_p2 );

SC\_METHOD(thread\_tmp\_15\_cast\_fu\_230\_p1);

sensitive << ( tmp\_12\_fu\_225\_p2 );

SC\_METHOD(thread\_tmp\_2\_cast\_fu\_167\_p1);

sensitive << ( j\_reg\_86 );

SC\_METHOD(thread\_tmp\_2\_fu\_171\_p2);

sensitive << ( tmp\_s\_reg\_259 );

sensitive << ( tmp\_2\_cast\_fu\_167\_p1 );

SC\_METHOD(thread\_tmp\_4\_cast\_fu\_193\_p1);

sensitive << ( k\_reg\_110 );

SC\_METHOD(thread\_tmp\_4\_fu\_197\_p2);

sensitive << ( tmp\_s\_reg\_259 );

sensitive << ( tmp\_4\_cast\_fu\_193\_p1 );

SC\_METHOD(thread\_tmp\_9\_fu\_137\_p3);

sensitive << ( i\_reg\_75 );

SC\_METHOD(thread\_tmp\_cast\_fu\_133\_p1);

sensitive << ( i\_reg\_75 );

SC\_METHOD(thread\_tmp\_s\_fu\_149\_p2);

sensitive << ( p\_shl\_cast\_fu\_145\_p1 );

sensitive << ( tmp\_cast\_fu\_133\_p1 );

SC\_METHOD(thread\_ap\_NS\_fsm);

sensitive << ( ap\_start );

sensitive << ( ap\_CS\_fsm );

sensitive << ( ap\_CS\_fsm\_state1 );

sensitive << ( ap\_CS\_fsm\_state2 );

sensitive << ( exitcond2\_fu\_121\_p2 );

sensitive << ( ap\_CS\_fsm\_state3 );

sensitive << ( exitcond1\_fu\_155\_p2 );

sensitive << ( ap\_CS\_fsm\_state4 );

sensitive << ( exitcond\_fu\_181\_p2 );

SC\_THREAD(thread\_hdltv\_gen);

sensitive << ( ap\_clk.pos() );

ap\_CS\_fsm = "00001";

static int apTFileNum = 0;

stringstream apTFilenSS;

apTFilenSS << "matrixmul\_sc\_trace\_" << apTFileNum ++;

string apTFn = apTFilenSS.str();

mVcdFile = sc\_create\_vcd\_trace\_file(apTFn.c\_str());

mVcdFile->set\_time\_unit(1, SC\_PS);

if (1) {

#ifdef \_\_HLS\_TRACE\_LEVEL\_PORT\_\_

sc\_trace(mVcdFile, ap\_clk, "(port)ap\_clk");

sc\_trace(mVcdFile, ap\_rst, "(port)ap\_rst");

sc\_trace(mVcdFile, ap\_start, "(port)ap\_start");

sc\_trace(mVcdFile, ap\_done, "(port)ap\_done");

sc\_trace(mVcdFile, ap\_idle, "(port)ap\_idle");

sc\_trace(mVcdFile, ap\_ready, "(port)ap\_ready");

sc\_trace(mVcdFile, a\_address0, "(port)a\_address0");

sc\_trace(mVcdFile, a\_ce0, "(port)a\_ce0");

sc\_trace(mVcdFile, a\_q0, "(port)a\_q0");

sc\_trace(mVcdFile, b\_address0, "(port)b\_address0");

sc\_trace(mVcdFile, b\_ce0, "(port)b\_ce0");

sc\_trace(mVcdFile, b\_q0, "(port)b\_q0");

sc\_trace(mVcdFile, res\_address0, "(port)res\_address0");

sc\_trace(mVcdFile, res\_ce0, "(port)res\_ce0");

sc\_trace(mVcdFile, res\_we0, "(port)res\_we0");

sc\_trace(mVcdFile, res\_d0, "(port)res\_d0");

#endif

#ifdef \_\_HLS\_TRACE\_LEVEL\_INT\_\_

sc\_trace(mVcdFile, ap\_CS\_fsm, "ap\_CS\_fsm");

sc\_trace(mVcdFile, ap\_CS\_fsm\_state1, "ap\_CS\_fsm\_state1");

sc\_trace(mVcdFile, i\_1\_fu\_127\_p2, "i\_1\_fu\_127\_p2");

sc\_trace(mVcdFile, i\_1\_reg\_254, "i\_1\_reg\_254");

sc\_trace(mVcdFile, ap\_CS\_fsm\_state2, "ap\_CS\_fsm\_state2");

sc\_trace(mVcdFile, tmp\_s\_fu\_149\_p2, "tmp\_s\_fu\_149\_p2");

sc\_trace(mVcdFile, tmp\_s\_reg\_259, "tmp\_s\_reg\_259");

sc\_trace(mVcdFile, exitcond2\_fu\_121\_p2, "exitcond2\_fu\_121\_p2");

sc\_trace(mVcdFile, j\_1\_fu\_161\_p2, "j\_1\_fu\_161\_p2");

sc\_trace(mVcdFile, j\_1\_reg\_268, "j\_1\_reg\_268");

sc\_trace(mVcdFile, ap\_CS\_fsm\_state3, "ap\_CS\_fsm\_state3");

sc\_trace(mVcdFile, tmp\_2\_cast\_fu\_167\_p1, "tmp\_2\_cast\_fu\_167\_p1");

sc\_trace(mVcdFile, tmp\_2\_cast\_reg\_273, "tmp\_2\_cast\_reg\_273");

sc\_trace(mVcdFile, exitcond1\_fu\_155\_p2, "exitcond1\_fu\_155\_p2");

sc\_trace(mVcdFile, res\_addr\_reg\_278, "res\_addr\_reg\_278");

sc\_trace(mVcdFile, k\_1\_fu\_187\_p2, "k\_1\_fu\_187\_p2");

sc\_trace(mVcdFile, k\_1\_reg\_286, "k\_1\_reg\_286");

sc\_trace(mVcdFile, ap\_CS\_fsm\_state4, "ap\_CS\_fsm\_state4");

sc\_trace(mVcdFile, exitcond\_fu\_181\_p2, "exitcond\_fu\_181\_p2");

sc\_trace(mVcdFile, grp\_fu\_243\_p3, "grp\_fu\_243\_p3");

sc\_trace(mVcdFile, ap\_CS\_fsm\_state5, "ap\_CS\_fsm\_state5");

sc\_trace(mVcdFile, i\_reg\_75, "i\_reg\_75");

sc\_trace(mVcdFile, j\_reg\_86, "j\_reg\_86");

sc\_trace(mVcdFile, res\_load\_reg\_97, "res\_load\_reg\_97");

sc\_trace(mVcdFile, k\_reg\_110, "k\_reg\_110");

sc\_trace(mVcdFile, tmp\_11\_cast\_fu\_176\_p1, "tmp\_11\_cast\_fu\_176\_p1");

sc\_trace(mVcdFile, tmp\_12\_cast\_fu\_202\_p1, "tmp\_12\_cast\_fu\_202\_p1");

sc\_trace(mVcdFile, tmp\_15\_cast\_fu\_230\_p1, "tmp\_15\_cast\_fu\_230\_p1");

sc\_trace(mVcdFile, tmp\_9\_fu\_137\_p3, "tmp\_9\_fu\_137\_p3");

sc\_trace(mVcdFile, p\_shl\_cast\_fu\_145\_p1, "p\_shl\_cast\_fu\_145\_p1");

sc\_trace(mVcdFile, tmp\_cast\_fu\_133\_p1, "tmp\_cast\_fu\_133\_p1");

sc\_trace(mVcdFile, tmp\_2\_fu\_171\_p2, "tmp\_2\_fu\_171\_p2");

sc\_trace(mVcdFile, tmp\_4\_cast\_fu\_193\_p1, "tmp\_4\_cast\_fu\_193\_p1");

sc\_trace(mVcdFile, tmp\_4\_fu\_197\_p2, "tmp\_4\_fu\_197\_p2");

sc\_trace(mVcdFile, tmp\_10\_fu\_207\_p3, "tmp\_10\_fu\_207\_p3");

sc\_trace(mVcdFile, p\_shl1\_cast\_fu\_215\_p1, "p\_shl1\_cast\_fu\_215\_p1");

sc\_trace(mVcdFile, tmp\_11\_fu\_219\_p2, "tmp\_11\_fu\_219\_p2");

sc\_trace(mVcdFile, tmp\_12\_fu\_225\_p2, "tmp\_12\_fu\_225\_p2");

sc\_trace(mVcdFile, ap\_NS\_fsm, "ap\_NS\_fsm");

#endif

}

mHdltvinHandle.open("matrixmul.hdltvin.dat");

mHdltvoutHandle.open("matrixmul.hdltvout.dat");

}

matrixmul::~matrixmul() {

if (mVcdFile)

sc\_close\_vcd\_trace\_file(mVcdFile);

mHdltvinHandle << "] " << endl;

mHdltvoutHandle << "] " << endl;

mHdltvinHandle.close();

mHdltvoutHandle.close();

delete matrixmul\_mac\_mulbkb\_U1;

}

void matrixmul::thread\_ap\_clk\_no\_reset\_() {

if ( ap\_rst.read() == ap\_const\_logic\_1) {

ap\_CS\_fsm = ap\_ST\_fsm\_state1;

} else {

ap\_CS\_fsm = ap\_NS\_fsm.read();

}

if ((esl\_seteq<1,1,1>(ap\_const\_logic\_1, ap\_CS\_fsm\_state3.read()) &&

esl\_seteq<1,1,1>(exitcond1\_fu\_155\_p2.read(), ap\_const\_lv1\_1))) {

i\_reg\_75 = i\_1\_reg\_254.read();

} else if ((esl\_seteq<1,1,1>(ap\_const\_logic\_1, ap\_CS\_fsm\_state1.read()) &&

esl\_seteq<1,1,1>(ap\_start.read(), ap\_const\_logic\_1))) {

i\_reg\_75 = ap\_const\_lv2\_0;

}

if ((esl\_seteq<1,1,1>(ap\_const\_logic\_1, ap\_CS\_fsm\_state4.read()) &&

esl\_seteq<1,1,1>(exitcond\_fu\_181\_p2.read(), ap\_const\_lv1\_1))) {

j\_reg\_86 = j\_1\_reg\_268.read();

} else if ((esl\_seteq<1,1,1>(ap\_const\_logic\_1, ap\_CS\_fsm\_state2.read()) &&

esl\_seteq<1,1,1>(exitcond2\_fu\_121\_p2.read(), ap\_const\_lv1\_0))) {

j\_reg\_86 = ap\_const\_lv2\_0;

}

if (esl\_seteq<1,1,1>(ap\_const\_logic\_1, ap\_CS\_fsm\_state5.read())) {

k\_reg\_110 = k\_1\_reg\_286.read();

} else if ((esl\_seteq<1,1,1>(ap\_const\_logic\_1, ap\_CS\_fsm\_state3.read()) &&

esl\_seteq<1,1,1>(ap\_const\_lv1\_0, exitcond1\_fu\_155\_p2.read()))) {

k\_reg\_110 = ap\_const\_lv2\_0;

}

if (esl\_seteq<1,1,1>(ap\_const\_logic\_1, ap\_CS\_fsm\_state5.read())) {

res\_load\_reg\_97 = grp\_fu\_243\_p3.read();

} else if ((esl\_seteq<1,1,1>(ap\_const\_logic\_1, ap\_CS\_fsm\_state3.read()) &&

esl\_seteq<1,1,1>(ap\_const\_lv1\_0, exitcond1\_fu\_155\_p2.read()))) {

res\_load\_reg\_97 = ap\_const\_lv16\_0;

}

if (esl\_seteq<1,1,1>(ap\_const\_logic\_1, ap\_CS\_fsm\_state2.read())) {

i\_1\_reg\_254 = i\_1\_fu\_127\_p2.read();

}

if (esl\_seteq<1,1,1>(ap\_const\_logic\_1, ap\_CS\_fsm\_state3.read())) {

j\_1\_reg\_268 = j\_1\_fu\_161\_p2.read();

}

if (esl\_seteq<1,1,1>(ap\_const\_logic\_1, ap\_CS\_fsm\_state4.read())) {

k\_1\_reg\_286 = k\_1\_fu\_187\_p2.read();

}

if ((esl\_seteq<1,1,1>(ap\_const\_logic\_1, ap\_CS\_fsm\_state3.read()) && esl\_seteq<1,1,1>(ap\_const\_lv1\_0, exitcond1\_fu\_155\_p2.read()))) {

res\_addr\_reg\_278 = (sc\_lv<4>) (tmp\_11\_cast\_fu\_176\_p1.read());

tmp\_2\_cast\_reg\_273 = tmp\_2\_cast\_fu\_167\_p1.read();

}

if ((esl\_seteq<1,1,1>(ap\_const\_logic\_1, ap\_CS\_fsm\_state2.read()) && esl\_seteq<1,1,1>(exitcond2\_fu\_121\_p2.read(), ap\_const\_lv1\_0))) {

tmp\_s\_reg\_259 = tmp\_s\_fu\_149\_p2.read();

}

}

void matrixmul::thread\_a\_address0() {

a\_address0 = (sc\_lv<4>) (tmp\_12\_cast\_fu\_202\_p1.read());

}

void matrixmul::thread\_a\_ce0() {

if (esl\_seteq<1,1,1>(ap\_const\_logic\_1, ap\_CS\_fsm\_state4.read())) {

a\_ce0 = ap\_const\_logic\_1;

} else {

a\_ce0 = ap\_const\_logic\_0;

}

}

void matrixmul::thread\_ap\_CS\_fsm\_state1() {

ap\_CS\_fsm\_state1 = ap\_CS\_fsm.read()[0];

}

void matrixmul::thread\_ap\_CS\_fsm\_state2() {

ap\_CS\_fsm\_state2 = ap\_CS\_fsm.read()[1];

}

void matrixmul::thread\_ap\_CS\_fsm\_state3() {

ap\_CS\_fsm\_state3 = ap\_CS\_fsm.read()[2];

}

void matrixmul::thread\_ap\_CS\_fsm\_state4() {

ap\_CS\_fsm\_state4 = ap\_CS\_fsm.read()[3];

}

void matrixmul::thread\_ap\_CS\_fsm\_state5() {

ap\_CS\_fsm\_state5 = ap\_CS\_fsm.read()[4];

}

void matrixmul::thread\_ap\_done() {

if ((esl\_seteq<1,1,1>(ap\_const\_logic\_1, ap\_CS\_fsm\_state2.read()) &&

esl\_seteq<1,1,1>(exitcond2\_fu\_121\_p2.read(), ap\_const\_lv1\_1))) {

ap\_done = ap\_const\_logic\_1;

} else {

ap\_done = ap\_const\_logic\_0;

}

}

void matrixmul::thread\_ap\_idle() {

if ((esl\_seteq<1,1,1>(ap\_const\_logic\_0, ap\_start.read()) &&

esl\_seteq<1,1,1>(ap\_const\_logic\_1, ap\_CS\_fsm\_state1.read()))) {

ap\_idle = ap\_const\_logic\_1;

} else {

ap\_idle = ap\_const\_logic\_0;

}

}

void matrixmul::thread\_ap\_ready() {

if ((esl\_seteq<1,1,1>(ap\_const\_logic\_1, ap\_CS\_fsm\_state2.read()) &&

esl\_seteq<1,1,1>(exitcond2\_fu\_121\_p2.read(), ap\_const\_lv1\_1))) {

ap\_ready = ap\_const\_logic\_1;

} else {

ap\_ready = ap\_const\_logic\_0;

}

}

void matrixmul::thread\_b\_address0() {

b\_address0 = (sc\_lv<4>) (tmp\_15\_cast\_fu\_230\_p1.read());

}

void matrixmul::thread\_b\_ce0() {

if (esl\_seteq<1,1,1>(ap\_const\_logic\_1, ap\_CS\_fsm\_state4.read())) {

b\_ce0 = ap\_const\_logic\_1;

} else {

b\_ce0 = ap\_const\_logic\_0;

}

}

void matrixmul::thread\_exitcond1\_fu\_155\_p2() {

exitcond1\_fu\_155\_p2 = (!j\_reg\_86.read().is\_01() || !ap\_const\_lv2\_3.is\_01())? sc\_lv<1>(): sc\_lv<1>(j\_reg\_86.read() == ap\_const\_lv2\_3);

}

void matrixmul::thread\_exitcond2\_fu\_121\_p2() {

exitcond2\_fu\_121\_p2 = (!i\_reg\_75.read().is\_01() || !ap\_const\_lv2\_3.is\_01())? sc\_lv<1>(): sc\_lv<1>(i\_reg\_75.read() == ap\_const\_lv2\_3);

}

void matrixmul::thread\_exitcond\_fu\_181\_p2() {

exitcond\_fu\_181\_p2 = (!k\_reg\_110.read().is\_01() || !ap\_const\_lv2\_3.is\_01())? sc\_lv<1>(): sc\_lv<1>(k\_reg\_110.read() == ap\_const\_lv2\_3);

}

void matrixmul::thread\_i\_1\_fu\_127\_p2() {

i\_1\_fu\_127\_p2 = (!i\_reg\_75.read().is\_01() || !ap\_const\_lv2\_1.is\_01())? sc\_lv<2>(): (sc\_biguint<2>(i\_reg\_75.read()) + sc\_biguint<2>(ap\_const\_lv2\_1));

}

void matrixmul::thread\_j\_1\_fu\_161\_p2() {

j\_1\_fu\_161\_p2 = (!j\_reg\_86.read().is\_01() || !ap\_const\_lv2\_1.is\_01())? sc\_lv<2>(): (sc\_biguint<2>(j\_reg\_86.read()) + sc\_biguint<2>(ap\_const\_lv2\_1));

}

void matrixmul::thread\_k\_1\_fu\_187\_p2() {

k\_1\_fu\_187\_p2 = (!k\_reg\_110.read().is\_01() || !ap\_const\_lv2\_1.is\_01())? sc\_lv<2>(): (sc\_biguint<2>(k\_reg\_110.read()) + sc\_biguint<2>(ap\_const\_lv2\_1));

}

void matrixmul::thread\_p\_shl1\_cast\_fu\_215\_p1() {

p\_shl1\_cast\_fu\_215\_p1 = esl\_zext<5,4>(tmp\_10\_fu\_207\_p3.read());

}

void matrixmul::thread\_p\_shl\_cast\_fu\_145\_p1() {

p\_shl\_cast\_fu\_145\_p1 = esl\_zext<5,4>(tmp\_9\_fu\_137\_p3.read());

}

void matrixmul::thread\_res\_address0() {

res\_address0 = res\_addr\_reg\_278.read();

}

void matrixmul::thread\_res\_ce0() {

if (esl\_seteq<1,1,1>(ap\_const\_logic\_1, ap\_CS\_fsm\_state4.read())) {

res\_ce0 = ap\_const\_logic\_1;

} else {

res\_ce0 = ap\_const\_logic\_0;

}

}

void matrixmul::thread\_res\_d0() {

res\_d0 = res\_load\_reg\_97.read();

}

void matrixmul::thread\_res\_we0() {

if (esl\_seteq<1,1,1>(ap\_const\_logic\_1, ap\_CS\_fsm\_state4.read())) {

res\_we0 = ap\_const\_logic\_1;

} else {

res\_we0 = ap\_const\_logic\_0;

}

}

void matrixmul::thread\_tmp\_10\_fu\_207\_p3() {

tmp\_10\_fu\_207\_p3 = esl\_concat<2,2>(k\_reg\_110.read(), ap\_const\_lv2\_0);

}

void matrixmul::thread\_tmp\_11\_cast\_fu\_176\_p1() {

tmp\_11\_cast\_fu\_176\_p1 = esl\_sext<64,5>(tmp\_2\_fu\_171\_p2.read());

}

void matrixmul::thread\_tmp\_11\_fu\_219\_p2() {

tmp\_11\_fu\_219\_p2 = (!p\_shl1\_cast\_fu\_215\_p1.read().is\_01() || !tmp\_4\_cast\_fu\_193\_p1.read().is\_01())? sc\_lv<5>(): (sc\_biguint<5>(p\_shl1\_cast\_fu\_215\_p1.read()) - sc\_biguint<5>(tmp\_4\_cast\_fu\_193\_p1.read()));

}

void matrixmul::thread\_tmp\_12\_cast\_fu\_202\_p1() {

tmp\_12\_cast\_fu\_202\_p1 = esl\_sext<64,5>(tmp\_4\_fu\_197\_p2.read());

}

void matrixmul::thread\_tmp\_12\_fu\_225\_p2() {

tmp\_12\_fu\_225\_p2 = (!tmp\_11\_fu\_219\_p2.read().is\_01() || !tmp\_2\_cast\_reg\_273.read().is\_01())? sc\_lv<5>(): (sc\_biguint<5>(tmp\_11\_fu\_219\_p2.read()) + sc\_biguint<5>(tmp\_2\_cast\_reg\_273.read()));

}

void matrixmul::thread\_tmp\_15\_cast\_fu\_230\_p1() {

tmp\_15\_cast\_fu\_230\_p1 = esl\_sext<64,5>(tmp\_12\_fu\_225\_p2.read());

}

void matrixmul::thread\_tmp\_2\_cast\_fu\_167\_p1() {

tmp\_2\_cast\_fu\_167\_p1 = esl\_zext<5,2>(j\_reg\_86.read());

}

void matrixmul::thread\_tmp\_2\_fu\_171\_p2() {

tmp\_2\_fu\_171\_p2 = (!tmp\_s\_reg\_259.read().is\_01() || !tmp\_2\_cast\_fu\_167\_p1.read().is\_01())? sc\_lv<5>(): (sc\_biguint<5>(tmp\_s\_reg\_259.read()) + sc\_biguint<5>(tmp\_2\_cast\_fu\_167\_p1.read()));

}

void matrixmul::thread\_tmp\_4\_cast\_fu\_193\_p1() {

tmp\_4\_cast\_fu\_193\_p1 = esl\_zext<5,2>(k\_reg\_110.read());

}

void matrixmul::thread\_tmp\_4\_fu\_197\_p2() {

tmp\_4\_fu\_197\_p2 = (!tmp\_s\_reg\_259.read().is\_01() || !tmp\_4\_cast\_fu\_193\_p1.read().is\_01())? sc\_lv<5>(): (sc\_biguint<5>(tmp\_s\_reg\_259.read()) + sc\_biguint<5>(tmp\_4\_cast\_fu\_193\_p1.read()));

}

void matrixmul::thread\_tmp\_9\_fu\_137\_p3() {

tmp\_9\_fu\_137\_p3 = esl\_concat<2,2>(i\_reg\_75.read(), ap\_const\_lv2\_0);

}

void matrixmul::thread\_tmp\_cast\_fu\_133\_p1() {

tmp\_cast\_fu\_133\_p1 = esl\_zext<5,2>(i\_reg\_75.read());

}

void matrixmul::thread\_tmp\_s\_fu\_149\_p2() {

tmp\_s\_fu\_149\_p2 = (!p\_shl\_cast\_fu\_145\_p1.read().is\_01() || !tmp\_cast\_fu\_133\_p1.read().is\_01())? sc\_lv<5>(): (sc\_biguint<5>(p\_shl\_cast\_fu\_145\_p1.read()) - sc\_biguint<5>(tmp\_cast\_fu\_133\_p1.read()));

}

void matrixmul::thread\_ap\_NS\_fsm() {

switch (ap\_CS\_fsm.read().to\_uint64()) {

case 1 :

if ((esl\_seteq<1,1,1>(ap\_const\_logic\_1, ap\_CS\_fsm\_state1.read()) && esl\_seteq<1,1,1>(ap\_start.read(), ap\_const\_logic\_1))) {

ap\_NS\_fsm = ap\_ST\_fsm\_state2;

} else {

ap\_NS\_fsm = ap\_ST\_fsm\_state1;

}

break;

case 2 :

if ((esl\_seteq<1,1,1>(ap\_const\_logic\_1, ap\_CS\_fsm\_state2.read()) && esl\_seteq<1,1,1>(exitcond2\_fu\_121\_p2.read(), ap\_const\_lv1\_1))) {

ap\_NS\_fsm = ap\_ST\_fsm\_state1;

} else {

ap\_NS\_fsm = ap\_ST\_fsm\_state3;

}

break;

case 4 :

if ((esl\_seteq<1,1,1>(ap\_const\_logic\_1, ap\_CS\_fsm\_state3.read()) && esl\_seteq<1,1,1>(exitcond1\_fu\_155\_p2.read(), ap\_const\_lv1\_1))) {

ap\_NS\_fsm = ap\_ST\_fsm\_state2;

} else {

ap\_NS\_fsm = ap\_ST\_fsm\_state4;

}

break;

case 8 :

if ((esl\_seteq<1,1,1>(ap\_const\_logic\_1, ap\_CS\_fsm\_state4.read()) && esl\_seteq<1,1,1>(exitcond\_fu\_181\_p2.read(), ap\_const\_lv1\_1))) {

ap\_NS\_fsm = ap\_ST\_fsm\_state3;

} else {

ap\_NS\_fsm = ap\_ST\_fsm\_state5;

}

break;

case 16 :

ap\_NS\_fsm = ap\_ST\_fsm\_state4;

break;

default :

ap\_NS\_fsm = "XXXXX";

break;

}

}

void matrixmul::thread\_hdltv\_gen() {

const char\* dump\_tv = std::getenv("AP\_WRITE\_TV");

if (!(dump\_tv && string(dump\_tv) == "on")) return;

wait();

mHdltvinHandle << "[ " << endl;

mHdltvoutHandle << "[ " << endl;

int ap\_cycleNo = 0;

while (1) {

wait();

const char\* mComma = ap\_cycleNo == 0 ? " " : ", " ;

mHdltvinHandle << mComma << "{" << " \"ap\_rst\" : \"" << ap\_rst.read() << "\" ";

mHdltvinHandle << " , " << " \"ap\_start\" : \"" << ap\_start.read() << "\" ";

mHdltvoutHandle << mComma << "{" << " \"ap\_done\" : \"" << ap\_done.read() << "\" ";

mHdltvoutHandle << " , " << " \"ap\_idle\" : \"" << ap\_idle.read() << "\" ";

mHdltvoutHandle << " , " << " \"ap\_ready\" : \"" << ap\_ready.read() << "\" ";

mHdltvoutHandle << " , " << " \"a\_address0\" : \"" << a\_address0.read() << "\" ";

mHdltvoutHandle << " , " << " \"a\_ce0\" : \"" << a\_ce0.read() << "\" ";

mHdltvinHandle << " , " << " \"a\_q0\" : \"" << a\_q0.read() << "\" ";

mHdltvoutHandle << " , " << " \"b\_address0\" : \"" << b\_address0.read() << "\" ";

mHdltvoutHandle << " , " << " \"b\_ce0\" : \"" << b\_ce0.read() << "\" ";

mHdltvinHandle << " , " << " \"b\_q0\" : \"" << b\_q0.read() << "\" ";

mHdltvoutHandle << " , " << " \"res\_address0\" : \"" << res\_address0.read() << "\" ";

mHdltvoutHandle << " , " << " \"res\_ce0\" : \"" << res\_ce0.read() << "\" ";

mHdltvoutHandle << " , " << " \"res\_we0\" : \"" << res\_we0.read() << "\" ";

mHdltvoutHandle << " , " << " \"res\_d0\" : \"" << res\_d0.read() << "\" ";

mHdltvinHandle << "}" << std::endl;

mHdltvoutHandle << "}" << std::endl;

ap\_cycleNo++;

}

}

}