Self-paced : 電子所 碩一 r09943113 董濟昀

atan2_cordic

1. Introduction

CORDIC(COordinate Rotation DIgital Computer)：只利用移位和加減運算計算常用三角函式值。在本例中，只要輸入任一組(x,y)，即可得到該座標與軸的夾角角度。

2. Csim / syn / co-sim

Csim：

top_atan2_csim.log

```
INFO: [SIM 2] *************** CSIM start ***************
INFO: [SIM 4] CSIM will launch GCC as the compiler.
   Compiling ../../../../cordic_test.cpp in debug mode
   Compiling ../../../../cordic_atan2.cpp in debug mode
   Generating csim.exe
INFO: [SIM 1] CSim done with 0 errors.
INFO: [SIM 3] *************** CSIM finish ***************

```

Syn：

Synthesis(solution2_cordic_bitaccurate)(top_atan2_csynth.rpt)

Summary

Clock	Target	Estimated	Uncertainty
ap_clk	2.50 ns	5.788 ns	0.31 ns

Latency

Summary

Latency (cycles)		Latency (absolute)		Interval (cycles)		
min	max	min	max	min	max	Type
2	15	11.576 ns	86.818 ns	2	15	none

Detail

Instance

Loop

Utilization Estimates

Summary

Name	BRAM_18K	DSP48E	FF	LUT	URAM
DSP	-	-	-	-	-
Expression	-	-	-	-	-
FIFO	-	-	-	-	-
Instance	0	-	190	596	-
Memory	-	-	-	-	-
Multiplexer	-	-	-	3	-
Register	-	-	3	-	-
Total	0	0	193	599	0
Available	1090	900	437200	218600	0
Utilization (%)	0	0	~0	~0	0

Interface

Summary

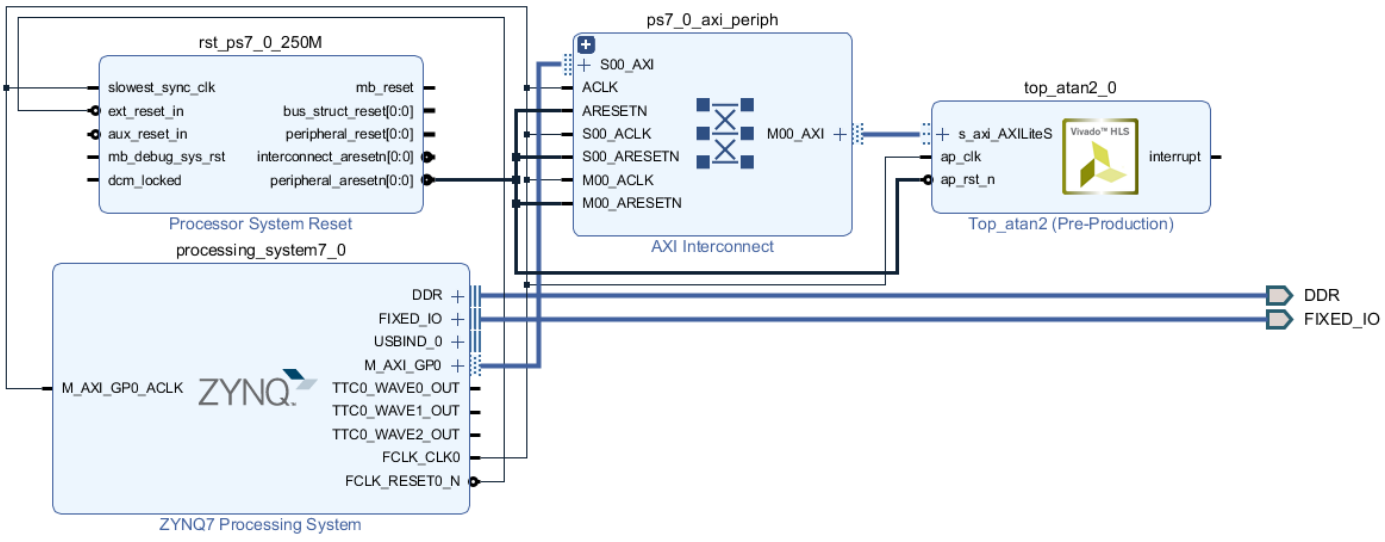
RTL Ports	Dir	Bits	Protocol	Source Object	C Type
ap_clk	in	1	ap_ctrl_hs	cordic_atan2	return value
ap_rst	in	1	ap_ctrl_hs	cordic_atan2	return value
ap_start	in	1	ap_ctrl_hs	cordic_atan2	return value
ap_done	out	1	ap_ctrl_hs	cordic_atan2	return value
ap_idle	out	1	ap_ctrl_hs	cordic_atan2	return value
ap_ready	out	1	ap_ctrl_hs	cordic_atan2	return value
y0_V	in	18	ap_none	y0_V	scalar
x0_V	in	18	ap_none	x0_V	scalar
zn_V	out	16	ap_vld	zn_V	pointer
zn_V_ap_vld	out	1	ap_vld	zn_V	pointer

Co-sim：

cosim.log

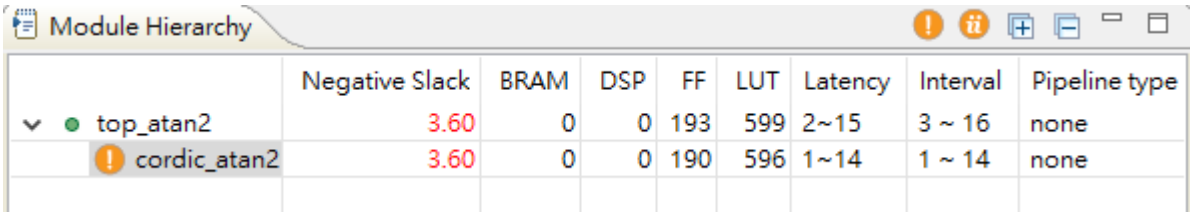
```
INFO: [COSIM-47] Using XSIM for RTL simulation.
INFO: [COSIM-14] Instrumenting C test bench ...
   Build using "C:/Xilinx/Vivado/2019.2/tps/win64/msys64/mingw64/bin/g++"
   Compiling apatb_top_atan2.cpp
   Compiling cordic_test.cpp_pre.cpp.tb.cpp
   Compiling cordic_atan2.cpp_pre.cpp.tb.cpp
   Generating cosim.tv.exe
INFO: [COSIM-302] Starting C TB testing ...
INFO: [COSIM-333] Generating C post check test bench ...
INFO: [COSIM-12] Generating RTL test bench ...
INFO: [COSIM-1] *** C/RTL co-simulation file generation completed. ***
INFO: [COSIM-323] Starting verilog simulation.
INFO: [COSIM-15] Starting XSIM ...
```

3. Implement



4. Optimize

   Original : time violated



	Negative Slack	BRAM	DSP	FF	LUT	Latency	Interval	Pipeline type
top_atan2	3.60	0	0	193	599	2~15	3 ~ 16	none
cordic_atan2	3.60	0	0	190	596	1~14	1 ~ 14	none

   sol 1: relax pipeline II

```
LOOP1: for (i = 0; i <= ROT; i++)
{

#pragma HLS PIPELINE II=2 //

    //dneg= (mode==1) ? (z<0) : (y>0);
    dneg = (y > 0);
    if (dneg) {
```

   sol 2: constant shift

```
//
const coord_t xsh[] = {x, x>>1, x>>2, x>>3, x>>4, x>>5, x>>6, x>>7, x>>8 };
const coord_t ysh[] = {y, y>>1, y>>2, y>>3, y>>4, y>>5, y>>6, y>>7, y>>8 };
```

```
#else
//          xp = x + y/lut_pow2[i]; //x+(y>>i);
//          yp = y - x/lut_pow2[i];//y-(x>>i);
//          xp = x + (y >> i); //x+(y>>i);
//          yp = y - (x >> i);//y-(x>>i);
            xp = x + ysh[i];
            yp = y - xsh[i];
#endif
```

5. Github

   https://github.com/jeff-tong/MSOC---Application-Acceleration-with-High-Level-Synthesis-

6. Reference

   https://www.itread01.com/content/1546286231.html