# HLS LabB Cholesky algorithm

## 111061627 陳佳詳

在線性代數中,cholesky decomposition 是指將一個正定的 Hermiton 矩陣分解成一個下三角矩陣,與其共軛轉置之乘積,下圖為他的公式。

If we write out the equation

$$\mathbf{A} = \mathbf{L}\mathbf{L}^T = egin{pmatrix} L_{11} & 0 & 0 \ L_{21} & L_{22} & 0 \ L_{31} & L_{32} & L_{33} \end{pmatrix} egin{pmatrix} L_{11} & L_{21} & L_{31} \ 0 & L_{22} & L_{32} \ 0 & 0 & L_{33} \end{pmatrix} = egin{pmatrix} L_{21}^2 & & ext{(symmetric)} \ L_{21}L_{11} & L_{21}^2 + L_{22}^2 \ L_{31}L_{11} & L_{31}L_{21} + L_{32}L_{22} & L_{31}^2 + L_{32}^2 + L_{33}^2 \end{pmatrix}$$

we obtain the following

$$\mathbf{L} = egin{pmatrix} \sqrt{A_{11}} & 0 & 0 \ A_{21}/L_{11} & \sqrt{A_{22}-L_{21}^2} & 0 \ A_{31}/L_{11} & \left(A_{32}-L_{31}L_{21}
ight)/L_{22} & \sqrt{A_{33}-L_{31}^2-L_{32}^2} \end{pmatrix}$$

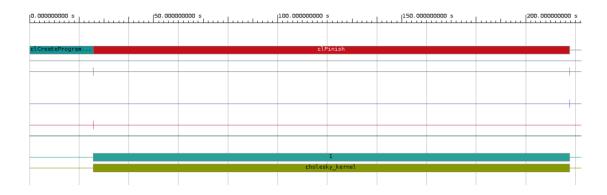
and therefore the following formulas for the entries of L:

$$egin{align} L_{j,j} &= (\pm) \sqrt{A_{j,j} - \sum_{k=1}^{j-1} L_{j,k}^2}, \ L_{i,j} &= rac{1}{L_{j,j}} \left(A_{i,j} - \sum_{k=1}^{j-1} L_{i,k} L_{j,k}
ight) \quad ext{for } i > j. \end{cases}$$

# 1. run this design in CPU

INFO: Matrix Row M: 512
INFO: Matrix Col N: 512
INFO: Finish CPU execution
INFO: CPU execution time is:15546 us
errA = 0
dataAN = 512
dataAM = 512
INFO: Result correct

### 2. module1 Baseline



Timing: \* Summary:

+	-+-		+-		+-		+
i Clock	i i	Targe	tΪ	Estimat	tedi	Uncertain	tyĺ
+	-+-		+-		+		+
lap_clk	i	3.33	nsİ	2.433	nsİ	0.90	nsİ
+	-+-		+-		+-		+

Latency: \* Summary:

Latency min	(cycles) max	Latency (	(absolute)     max	Interva	al İ Pipeli ax I Type	neİ İ
			? i	,		
++			+		+	+

### Utilization estimation:

# = Utilization Estimates

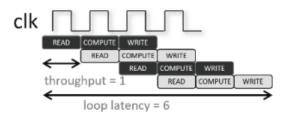
* Summary:					
Name	BRAM_18K	DSP	FF	LUT	URAM
IDSP  Expression  FIFO  Instance  Memory  Multiplexer  Register	-    -    -    4    32    -		3561 0 3561 0 1071	0   2144	-    0    0
Total	36	20	4632	5566	0
Available SLR	1344	2976	871680	435840	320
Utilization SLR (%)	2	~0	~0	1	0
Available	26881	5952	1743360	871680	640
Utilization (%)	+	~01	~0	~0 +	   0  

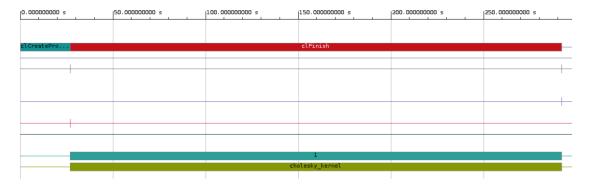
### Interface:

RTL Ports	Dir	Bits	Protocol	Source Object	С Туре
s_axi_control_AWVALID	+in		s_axi		scalar
s_axi_control_AWREADY	out		s_axi		
s_axi_control_AWADDR	in		s_axil	control	
s_axi_control_WVALID	l in		s_axil		l scaları I scaları
s_axi_control_WREADY s_axi_control_WDATA	l out		s_axil		scalari   scalari
s_axi_control_WSTRB	in		s_axil s_axil		scalari
s_axi_control_ARVALID	ini	4.1	s_axi		scalar
s_axi_control_ARREADY	out	4.1	s_axi		
s_axi_control_ARADDR	in		s_axi		
s axi control RVALID	out	11	s_axi		
s_axi_control_RREADY	l in		s_axi	control	scalar
s_axi_control_RDATA	out		s_axi		scalar
s_axi_control_RRESP	out		s_axi		
s_axi_control_BVALID	out		s_axil		
s_axi_control_BREADY	l in		s_axil		l scaları İ scaları
s_axi_control_BRESP ap_local_block	l out		s_axil ap_ctrl_chain		
ap_local_block	in		ap_ctrl_chain		return valuel
ap_rst_n	ini	7.1	ap_ctrl_chain		
interrupt	out		ap_ctrl_chain		
m_axi_gmen_AWVALID	out	11	n_axi		
m_axi_gmem_AWREADY	l in		m_axi	gmen	
m_axi_gmem_AWADDR	out		n_axi	_	
n_axi_gmen_AWID	out		m_axi!		
m_axi_gmem_AWLEN	out		n_axil	-	
m_axi_gmem_AWSIZE	out		n_axil		
n_axi_gmen_AWBURST n_axi_gmen_AWLOCK	l out		n_axil n_axil		
n_axi_gmen_AWCACHE	lout		n_axi		
n_axi_gmen_AWPROT	lout		n_axi	_	
n_axi_gmen_AWQOS	out		n_axil	_	
n_axi_gmen_AWREGION	out		n_axi∣	_	
m_axi_gmem_AWUSER	out		m_axil	gmen	pointer
m axi gmem VVALID	out	11	n axi	gmen	pointer
In axi gnen WREADY	inl	11	m axil	gmenl	pointer
lm_axi_gmem_WDATA	outl	641	m_axil	gmenl	pointer
lm_axi_gmem_WSTRB	outl	81	m_axil	gmenl	pointer
lm_axi_gmem_WLAST	outl	11	m_axil	gmenl	pointerl
n_axi_gnen_WID	out	11	m_axil	gmenl	pointer
lm_axi_gmem_WUSER	out	11	m_axil	gmenl	pointer
m_axi_gmem_ARVALID	outl	1 I 1 I	m_axil	gmenl	pointer
n_axi_gnen_ARREADY  n_axi_gnen_ARADDR	inl outl	641	m_axil m_axil	gmenl gmenl	pointer  pointer
Im_axi_gmem_ARID	out	11	m_axil	gmenl	pointer
lm_axi_gmem_ARLEN	outl	8i	m_axil	gmenl	pointer
lm_axi_gmem_ARSIZE	outl	31	m_axil	gmenl	pointer
lm_axi_gmem_ARBURST	outl	21	m_axil	gmenl	pointer
n_axi_gmem_ARLOCK	outl	21	m_axil	gmenl	pointerl
n_axi_gmem_ARCACHE	out	41	m_axil	gmen	pointer
lm_axi_gmem_ARPROT	out	31	m_axil	gmenl	pointer
lm_axi_gmem_ARQOS	out	41	m_axil	gmenl	pointer
m_axi_gmem_ARREGION	outl	41	m_axil	gmenl	pointer
n_axi_gmen_ARUSER	out! in!	1 I 1 I	m_axil m_axil	gmenl	pointer  pointer
In_axi_gmen_RREADY	outl	11	m_axil	gmenl gmenl	pointer
Im_axi_gmem_RDATA	inl	641	m_axil	gmenl	pointer
lm_axi_gmem_RLAST	ini	11	m_axil	gmenl	pointer
lm_axi_gmem_RID	inl	îi	m_axil	gnenl	pointer
lm_axi_gmem_RUSER	inl	11	m_axil	gmenl	pointer
m_axi_gmem_RRESP	inl	21	m_axil	gmenl	pointer
lm_axi_gmem_BVALID	inl	11	m_axil	gmenl	pointer
lm_axi_gmem_BREADY	outl	11	m_axil	gmenl	pointer
lm_axi_gmem_BRESP	inl	21	m_axil	gmenl	pointer
In_axi_gnen_BID   In_axi_gnen_BUSER	inl	1 I 1 I	m_axil	gmenl	pointer
iw_avi_Swew_doork	inl	11	m_axil	gmenl	pointer

# 3. module2 pipeline

This module is meant to focus on the pipeline pragma and go through the description below. The kernel source code with the loops annotated with the pragma will produce the same results as in module 1, that's because since simple loops and inner loops (for nested loops) are automatically pipelined by the tool.





#### Performance estimation:

```
+ Timing:
    * Summary:
    +-----+
    | Clock | Target | Estimated | Uncertainty|
    +-----+
    | lap_clk | 3.33 ns| 2.433 ns| 0.90 ns|
    +-----+
```

	Sunnary:			
		+		
		Latency     min		
		+		
Ī		?		
+-		+		

#### = Utilization Estimates

•	$\sim$					
•	×1	n n	ma	r	$\mathbf{v}$	۰
	~	U. JII	ma		J	٠

Name	BRAM_18K	DSP	FF	LUT	URAMI
IDSP  Expression  FIFO  Instance  Memory  Multiplexer  Register	-   -   -   4   32   -	-   -   20   -   -	-   01 -   3561 01 -   1135	01 21441	- I 01 01
Total	361	201	46961	55661	01
Available SLR	1344	2976	871680	4358401	320
Utilization SLR (%)	2	~0]	~0]	1	0
Available	2688	5952	1743360	871680	6401
Utilization (%)	11	~0	~0	~0	01

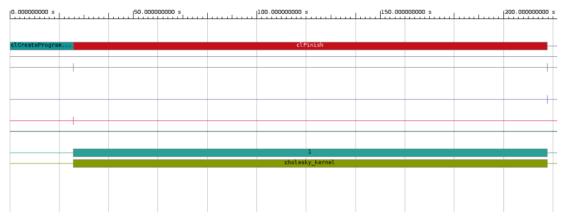
#### Interface:

+	++	+	+		+
RTL Ports	Dir I	Bits	Protocol	Source Object	C Type
+	++	+			+
s_axi_control_AWVALID	inl	11	s_axi	control	scalari
ls_axi_control_AVREADY	outl	11	s_axil		scalari
ls_axi_control_AVADDR	l inl	61	s_axi	control	scalari
ls_axi_control_WVALID	l in l	11	s_axil	control	scalari
ls_axi_control_VREADY	out		s_axi	control	scalari
s_axi_control_VDATA	l in l		s_axil	control	scalari
ls_axi_control_WSTRB	inl	41	s_axi	control	scalari
s_axi_control_ARVALID	l in l	11	s_axil	control	scalari
ls_axi_control_ARREADY	out		s_axi		scalar
s_axi_control_ARADDR	l in l	61	s_axil	control	
s_axi_control_RVALID	out	11	s_axi	control	scalar
s_axi_control_RREADY	l in l		s_axil	control	
ls_axi_control_RDATA	out	321	s_axi		scalar
s_axi_control_RRESP	out	21	s_axil	control	
ls_axi_control_BVALID	out		s_axi	control	scalari
ls_axi_control_BREADY	l in l		s_axil	control	scalari
ls_axi_control_BRESP	out	21	s_axi		scalari
lap_local_block	out	11	ap_ctrl_chain		
lap_clk	inl	11	ap_ctrl_chain		
ap_rst_n	l in l	11	ap_ctrl_chain		
linterrupt	l outl	11	ap ctrl chainl	cholesky kernell	return valuel

ln_axi_gmen0_AWVALID	out	11	n_axil	gnem01	pointerl
n_axi_gmenO_AWREADY	l in	11	m_axil	gnem01	pointer
n_axi_gmenO_AVADDR	out	641	m_axil	gmem01	pointer
n_axi_gmenO_AVID	out	11	m_axi	gmem01	pointer
ln_axi_gmenO_AVLEN	outl	81	m_axil	gmem01	pointer
n_axi_gmenO_AWSIZE	out	31	m_axil	gmem01	pointer
n axi gmenO AVBURST	out	21	m_axi	gmem01	pointer
ln_axi_gmenO_AVLOCK	out	21	m_axil	gmem01	pointerl
ln_axi_gmenO_AVCACHE	out	41	m_axil	gnem01	pointer
ln_axi_gmenO_AWPROT	l outl	31	m_axil	gmem01	pointerl
ln_axi_gmen0_AWQOS	out	41	m_axi	gnem01	pointerl
n_axi_gmenO_AWREGION	l outl	41	m_axil	gmem01	pointerl
n_axi_gmenO_AVUSER	out	11	m_axil	gmem01	pointerl
ln_axi_gmenO_WVALID	out	11	m_axil	gnem01	pointer
ln_axi_gmenO_WREADY	l inl	11	m_axil	gmem01	pointerl
lm axi gmemO WDATA	out	641	m_axil	gnem01	pointer
ln_axi_gmen0_WSTRB	out	81	m_axil	gmem01	pointerl
lm_axi_gmenO_WLAST	out	11	m_axil	gnem01	pointerl
ln_axi_gmenO_WID	out	11	m_axi	gnem01	pointerl
n axi gmenO WUSER	out	11	m_axi	gmem01	pointer
ln_axi_gmenO_ARVALID	out	11	m axil	gnem01	pointer
n axi gmenO ARREADY	l inl	11	m_axil	gmem01	pointerl
lm axi gmenO ARADDR	out	641	m axil	gnem01	pointer
ln_axi_gmenO_ARID	out	11	m_axil	gmem01	pointer
n_axi_gmenO_ARLEN	out	81	m_axi	gmem01	pointer
ln_axi_gmen0_ARSIZE	outl	31	m_axil	gmem01	pointerl
lm_axi_gmen0_ARBURST	out	21	m_axil	gmem01	pointerl
n_axi_gmenO_ARLOCK	out	21	m_axil	gmem01	pointer
n_axi_gmenO_ARCACHE	out	41	m_axil	gmem01	pointerl
lm axi gmemO ARPROT	out	31	m_axil	gnem01	pointer
ln_axi_gmen0_ARQOS	out	41	m_axil	gmem01	pointerl
n_axi_gmenO_ARREGION	out	41	m_axi	gnem01	pointer
n_axi_gmenO_ARUSER	l outl	11	m_axi	gmem01	pointerl
n_axi_gmenO_RVALID	l inl	11	m_axi	gmem01	pointer
lm axi gmemO RREADY	out	11	m_axi	gmem01	pointer
ln_axi_gmenO_RDATA	l in	641	m_axi	gmem01	pointer
ln_axi_gmenO_RLAST	l inl	11	m_axi	gnem01	pointer
n_axi_gmenO_RID	l inl	11	m_axi	gmem01	pointerl
n_axi_gmenO_RUSER	l in	11	m_axil	gnem01	pointer
ln_axi_gmen0_RRESP	l inl	21	m_axil	gmem01	pointer
n_axi_gmenO_BVALID	l in	11	m_axil	gmem01	pointerl
l n_ax i_gmenO_BREADY	outl	11	m_axil	gmem01	pointer
lm_axi_gmenO_BRESP	in	21	m_axil	gmem01	pointerl
ln_axi_gmenO_BID	l inl	11	m_axil	gmem01	pointer
ln_axi_gmenO_BUSER	l in	11	n_axil	gmem01	pointerl
				-	

### 4. module 3 datatype

In this module both the kernel and host code are modified to use 32-bit floating point data types(float) instead of the 64-bit floating point(double) to show the performance and Xilinx utilization beneficial impact of downsizing data types.



#### = Performance Estimates

+ Timing: \* Summary:

+	-+-		+	+-	+
Clock		Target	Estima	tedl	Uncertaintyl
+	+-		+	+-	+
lap_clk	i	3.33 ns	i 2.433	nsİ	0.90 ກຮໄ
+	+-		+	+-	+

# + Latency: \* Summary:

+	+	+	+	+	+	++
Laten	cy (cycles	s)   Late	ncy (abs	olute)	Interva	l   Pipelinel
						х І Туре І
+	+	+	+	+	+	++
1	?1	?1	?1	?1	?1	?l nol
+	+	+	+	+	+	++

#### — Utilization Estimates

*	Summary:	

4	L		L	L	L <b>_</b>
Name	BRAM_18K	DSP	FF	LUT	URAMI
+	-  -  2 16   -		2496 0 2496 0 - 930	0     1820	-    0    0
Total	18	14	3426	4502	0
Available SLR	1344			435840	320
Utilization SLR (%)	1				0
Available	2688	5952	1743360	871680	640
Utilization (%)	~01	~0	~0	~01	01

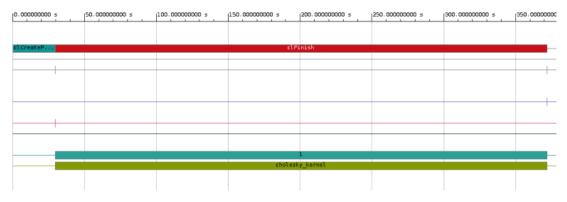
RTL Ports	Dir	Bits	Protocol I	Source Object	СТуре
s_axi_control_AWVALID	i in	11	s_axil	control	scalar
s_axi_control_AVREADY	out	11	s_axi	control	scalari
s_axi_control_AVADDR	l in l	61	s_axil	control	scalari
s_axi_control_WVALID	l in	11	s_axi	control	scalar
s_axi_control_WREADY	out	11	s_axi	control	scalar
s_axi_control_WDATA	l in	321	s_axi	controll	scalari
s_axi_control_WSTRB	l in	41	s_axil	control	scalari
s_axi_control_ARVALID	l in	11	s_axil	control	scalari
s_axi_control_ARREADY	out	11 61	s_axil	control	scalar  scalar
s_axi_control_ARADDR s_axi_control_RVALID	in    out	11	s_axil s_axil	controll controll	scalari
s_axi_control_RREADY	l inl	ii	s_axil	control	scalari
s_axi_control_RDATA	out	321	s_axil	control	scalar
s_axi_control_RRESP	outl	21	s_axil	control	scalari
s_axi_control_BVALID	outl	ĩi	s_axil	control	scalari
s_axi_control_BREADY	inl	1i	s_axil	control	scalari
s_axi_control_BRESP	lout	21	s_axi	control	scalar
ap_local_block	out	11	ap ctrl chain	cholesky_kernel	return value
ap_clk	l inl	11	ap_ctrl_chain	cholesky_kernel	return value
ap_rst_n	l in l	11	ap_ctrl_chain	cholesky_kernel	return valuel
interrupt	out	11	ap_ctrl_chain	cholesky_kernel	return value
n_axi_gmemO_AWVALID	out	11	m_axi	gnen0	pointer
m_axi_gmemO_AVREADY	l in	-11	m_axil	gnen0	pointer
m_axi_gmemO_AWADDR	out	641	m_axil	gnen0	pointer
m_axi_gmemO_AVID	out	11	m_axil	gnen0	pointer
n_axi_gmemO_AVLEN	out	81	m_axil	gnen01	pointer
n_axi_gmemO_AWSIZE	out	31	n_axi!	gnen0	pointer
n_axi_gmemO_AVBURST	out	21	n_axil	gnen0	pointer
m_axi_gmemO_AVLOCK	out	21	m_axil	gnen01	pointer
m_axi_gmemO_AWCACHE	out	41	m_axil	gnen01	pointer
m_axi_gmemO_AWPROT	out	31	n_axil	gnen01	pointer
n_axi_gmemO_AVQOS n_axi_gmemO_AVREGION	out	41 41	n_axil	gnen0	pointer
n_axi_gmemO_AVUSER	out	11	n_axil n_axil	gnen0  gnen0	pointer  pointer
n_axi_gmemO_WVALID	out	11	n_axil	gnen01	pointer
n_axi_gmemO_VREADY	inl	ii	n_axil	gnenOl	pointer
n_axi_gmemO_VDATA	l outl	32i	n_axil	gnen01	pointer
n_axi_gmemO_VSTRB	outl	41	n_axil	gnen01	pointer
n_axi_gmemO_WLAST	out	ii	m_axi	gnenOl	pointer
n_axi_gmemO_VID	out	11	m_axil	gnen01	pointer
n_axi_gmemO_WUSER	out	11	m_axil	gnen01	pointer
m_axi_gmemO_ARVALID	out	11	m_axil	gnen01	pointer
m_axi_gmemO_ARREADY	l in l	11	m_axil	gnen01	pointerl
m_axi_gmemO_ARADDR	l outl	641	m_axil	gmen0	pointer
m_axi_gmemO_ARID	out	11	m_axil	gnen0	pointer
n_axi_gmemO_ARLEN	out	81	m_axil	gnen0	pointer
m_axi_gmemO_ARSIZE	out	31	m_axil	gmen0	pointer
m_axi_gmemO_ARBURST	out	21	m_axil	gnen01	pointer
m_axi_gmemO_ARLOCK	out	21	m_axil	gnen01	pointer
m_axi_gmemO_ARCACHE	out	41 31	m_axil	gnen0	pointer
m_axi_gmemO_ARPROT	out	31 41	m_axil	gnen0  gnen0	pointer  pointer
n_axi_gmemO_ARQOS n_axi_gmemO_ARREGION	out	41	m_axil	gnenol gnenol	pointer pointer
n_axi_gmemO_ARUSER	l outl	11	n_axil n_axil	gnenor gnenOl	pointer
			_	•	
m_axi_gmemO_RVALID	l in	11	m_axi!	gmem01	pointer
n_axi_gmemO_RREADY	out	11	m_axil	gnem0	pointer
m_axi_gmemO_RDATA	l in	321	m_axil	gnem0	pointer
m_axi_gmemO_RLAST	l in	11	m_axil	gmem01	pointer
m_axi_gmemO_RID	l inl	11	m_axil	gnem01	pointer
m_axi_gmemO_RUSER	l inl	11	m_axil	gmem01	pointer
m_axi_gmem0_RRESP	l inl	21	m_axil	gmem01	pointer
m_axi_gmemO_BVALID	l inl	11	m_axil	gmem01	pointer
m_axi_gmemO_BREADY	out	11 21	m_axil	gmem01	pointer
n_axi_gmemO_BRESP n_axi_gmemO_BID	l inl	11	m_axil	gmem01	pointer
W_WXI_EMENO_DID	l inl	11	m_axi	gnem01 gnem01	pointer pointer
n_axi_gmemO_BUSER	1 10 1		m_axil	27 M G W L L	DOINTOY:

#### 5. module4 dataflow

The DATAFLOW pragma enables task-level pipelining, allowing functions and loops to overlap in their operation, increasing the concurrency of the register transfer level(RTL) implementation, and increasing the overall throughput of the design.

```
wr_loop_j: for (int j = 0; j < TILE_PER_ROW; ++j) {
    #pragma HLS DATAFLOW
    wr_buf_loop_m: for (int m = 0; m < HEIGHT; ++m) {
        wr_buf_loop_n: for (int n = 0; n < WIDTH; ++n) {
            #pragma HLS PIPELINE
            // should burst WIDTH in WORD beat
            outFifo >> tile[m][n];
        }
    }
    wr_loop_m: for (int m = 0; m < HEIGHT; ++m) {
        wr_loop_n: for (int n = 0; n < WIDTH; ++n) {
        #pragma HLS PIPELINE
            outx[HEIGHT*TILE_PER_ROW*WIDTH*i+TILE_PER_ROW*WIDTH*m+WIDTH*j+n] = tile[m][n];
        }
    }
}</pre>
```

#### 以下程式會呼叫 chol col 16 次



### = Performance Estimates

+ Timing: \* Summary:

++	+	+	+
i Clock i	Target İ	Estimatedi	Uncertainty
++	+	+	+
			0.90 ກຮໄ
++	+		+

# + Latency: \* Summary:

Latency min	(cycles)	Latency (	(absolute)	Inte	rval i	Pipelinel
	max	min	max	min 1	max l	Type I
	?	i ?	?i	?İ	?i	nol

### — Utilization Estimates

# \* Summary:

Name	BRAM_18K	DSP	FF	LUT	URAMI
IDSP  Expression  FIFO  Instance  Memory  Multiplexer  Register	-  -  4  32  -		81560 81560 0 0 - 631	01   31891	- I 0 I 64 I
Total	36	196	82191	63505	64
Available SLR	1344	2976	871680	435840	320
Utilization SLR (%)	21	6	9	14	201
Available	2688	5952	1743360	871680	640
Utilization (%)	11	3	4	7 7 1 1 7 1	101

* Summary:					
+	Dir	Bitsl		Source Object	
+	+   in	++		+	++
s_axi_control_AWREADY	out				
s_axi_control_AWADDR	in				
s_axi_control_WVALID	i in				
ls_axi_control_WREADY	out				
ls_axi_control_WDATA	l in	321			scalar
ls_axi_control_WSTRB	l in				
s_axi_control_ARVALID	l in				
s_axi_control_ARREADY	out				
s_axi_control_ARADDR	l in				scalar
ls_axi_control_RVALID ls_axi_control_RREADY	l out I in				l scaları I scaları
s_axi_control_RDATA	out		s_axi s_axi		
s_axi_control_RRESP	out		s_axi		
s_axi_control_BVALID	out				
ls_axi_control_BREADY	l in				
ls_axi_control_BRESP	out	21	s_axi		
lap_local_block	out	11			
[ap_clk	l in				
ap_rst_n	l in				
linterrupt	out				
lm_axi_gmenO_AWVALID	out				
ln_axi_gmenO_AVREADY ln_axi_gmenO_AVADDR	l in Lout				
In_axi_gmenO_AVID	lout				
In_axi_gmenO_AVLEN	out				
ln_axi_gmenO_AWSIZE	out				
ln_axi_gmenO_AVBURST	out		n_axi		
lm_axi_gmen0_AVLOCK	out	21	m_axi		
n_axi_gmenO_AVCACHE	out			gnem0	
lm_axi_gmenO_AWPROT	out				
n_axi_gmenO_AVQOS	out				
m_axi_gmenO_AVREGION	out				
lm_axi_gmenO_AVUSER lm_axi_gmenO_VVALID	l out lout				
In_axi_gmenO_VREADY	l in				
ln_axi_gmenO_VDATA	out				
lm_axi_gmenO_VSTRB	out				
ln_axi_gmen0_VLAST	out				
lm_axi_gmenO_VID	out	11	m_axi		pointerl
n_axi_gmen0_VUSER	out				
n_axi_gmenO_ARVALID	out				
m_axi_gmenO_ARREADY	in!	1 1	m_axi	gnem0	l pointer!
lm_axi_gmenO_ARADDR	out				
lm_axi_gmemO_ARID	out				
n_axi_gmenO_ARLEN	out				
m_axi_gmenO_ARSIZE	out				pointer
lm_axi_gmenO_ARBURST	out		m_axi		
ln_axi_gmenO_ARLOCK ln_axi_gmenO_ARCACHE	l out l out				
In_axi_gmenO_ARPROT	out				
lm_axi_gmenO_ARQOS	out				
lm_axi_gmenO_ARREGION	out				
lm_axi_gmenO_ARUSER	out	11			
lm_axi_gmenO_RVALID	l in				
lm_axi_gmenO_RREADY	out				
m_axi_gmenO_RDATA	l in				
lm_axi_gmen0_RLAST	l in				
lm_axi_gmenO_RID	l in				
ln_axi_gmenO_RUSER ln_axi_gmenO_RRESP	l in I in				
In_axi_gmenO_BVALID	l in				
lm_axi_gmenO_BREADY	out				
lm_axi_gmenO_BRESP	in				
lm_axi_gmenO_BID	i in	1	m_axi		
lm_axi_gmen0_BUSER	l in		m_axi		

# Result Summary:

Module	CPU	Module1	Moduel2	Module3	Module4
Exe. Time	21461	793950	793732	536784	11698
Speed up(cpu)	1	0.03x	0.03x	0.04x	1.83x
Speed up	N/A	1	1	1.48x	68x

https://github.com/jiaxianggg-chen/cholesky-algorithm