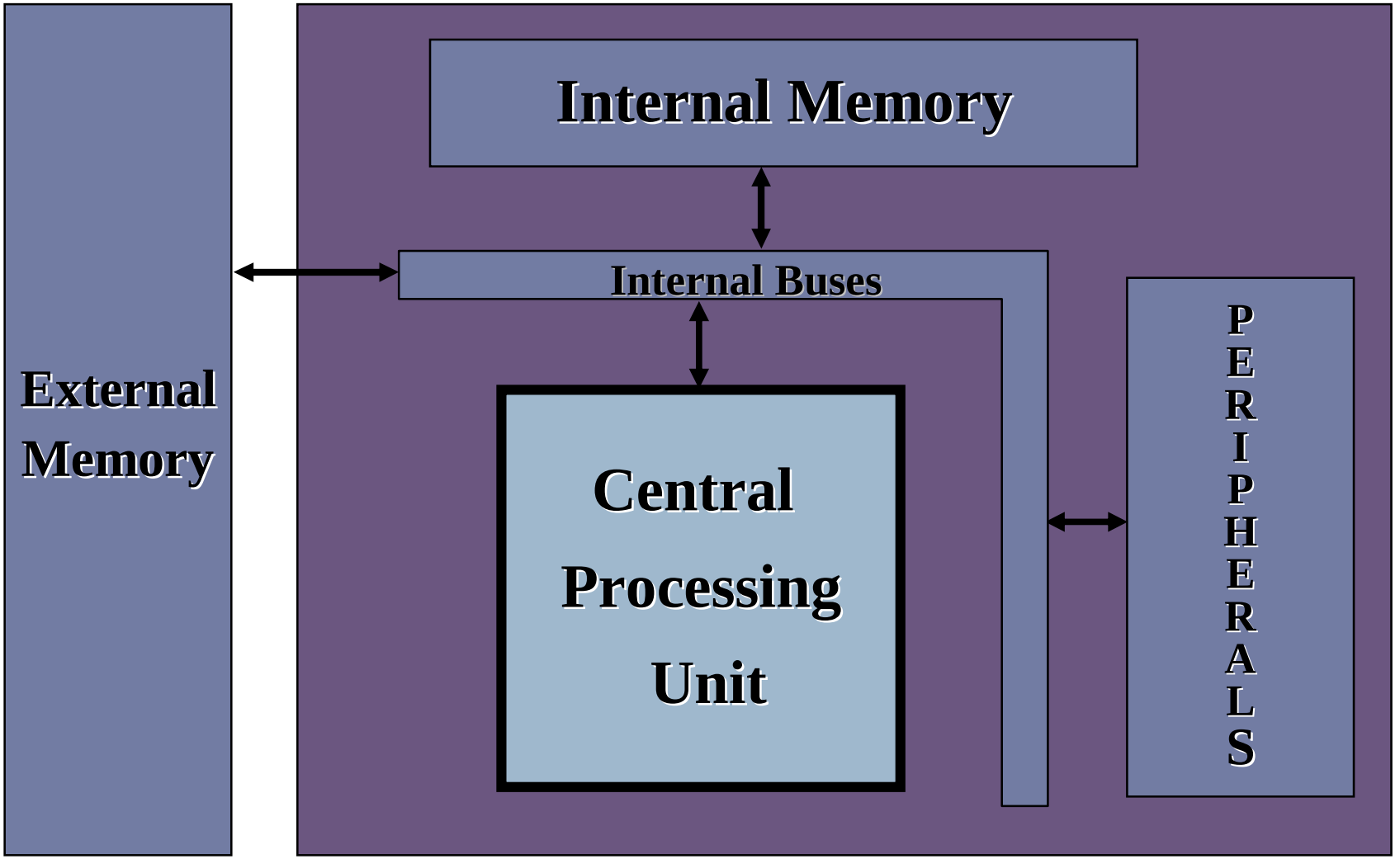
华清远见
FARSIGHT
嵌入式培训专家

RedLogic
华清远见旗下品牌

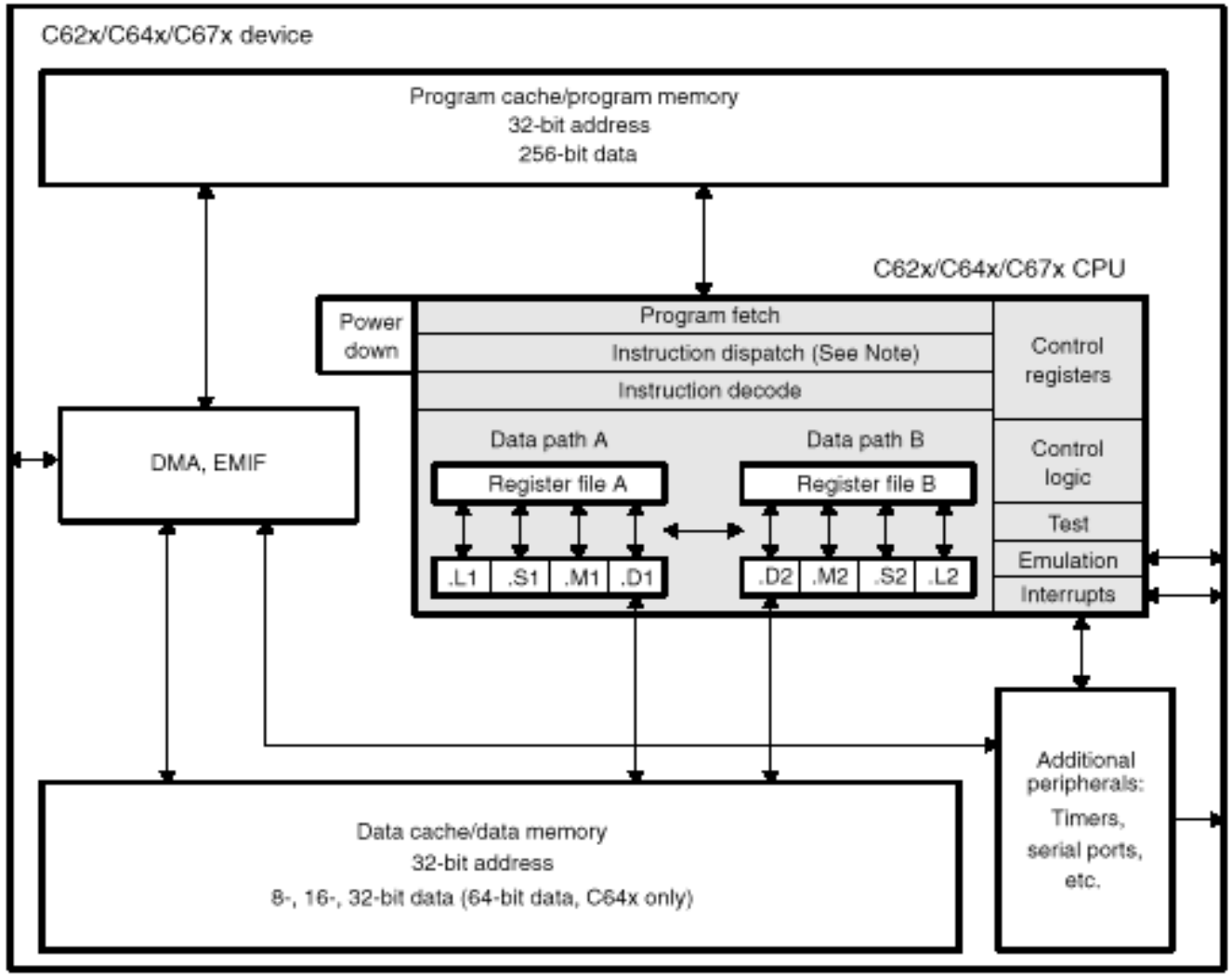
C6000 体系结构和汇编语言

General DSP System Block Diagram

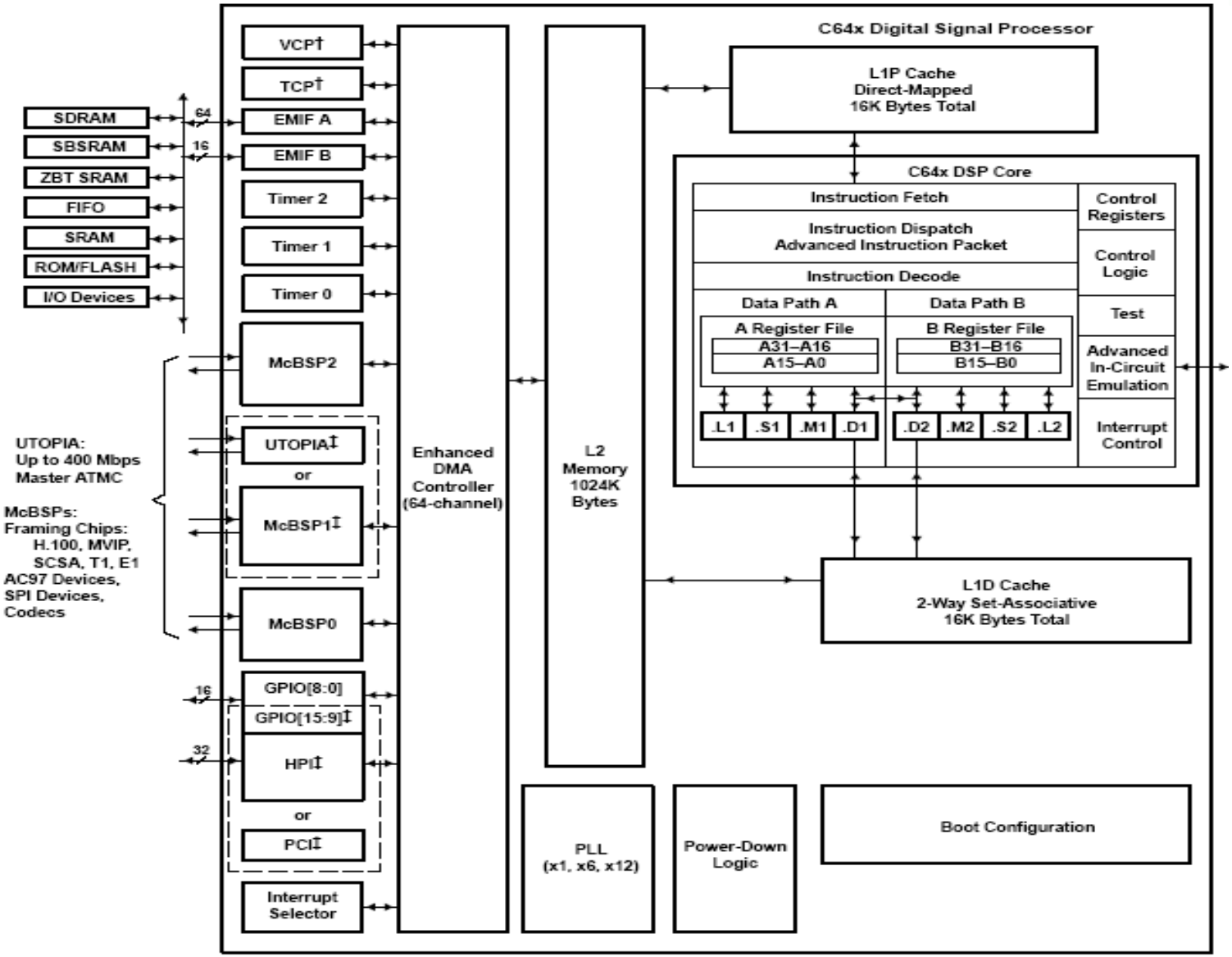


C6000 CPU Architecture

VLIW, Very Long Instruction word

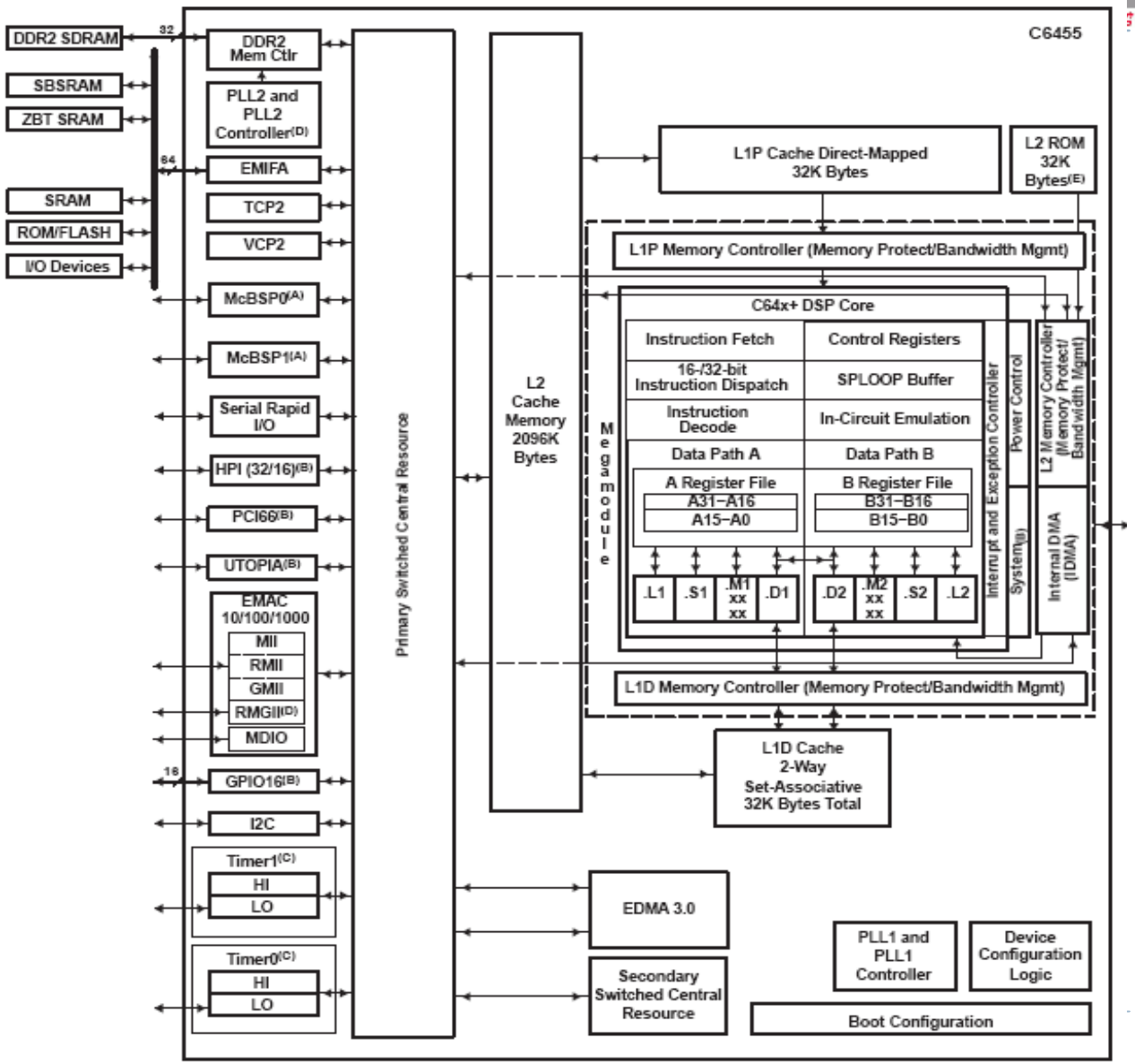


TMS320C6416

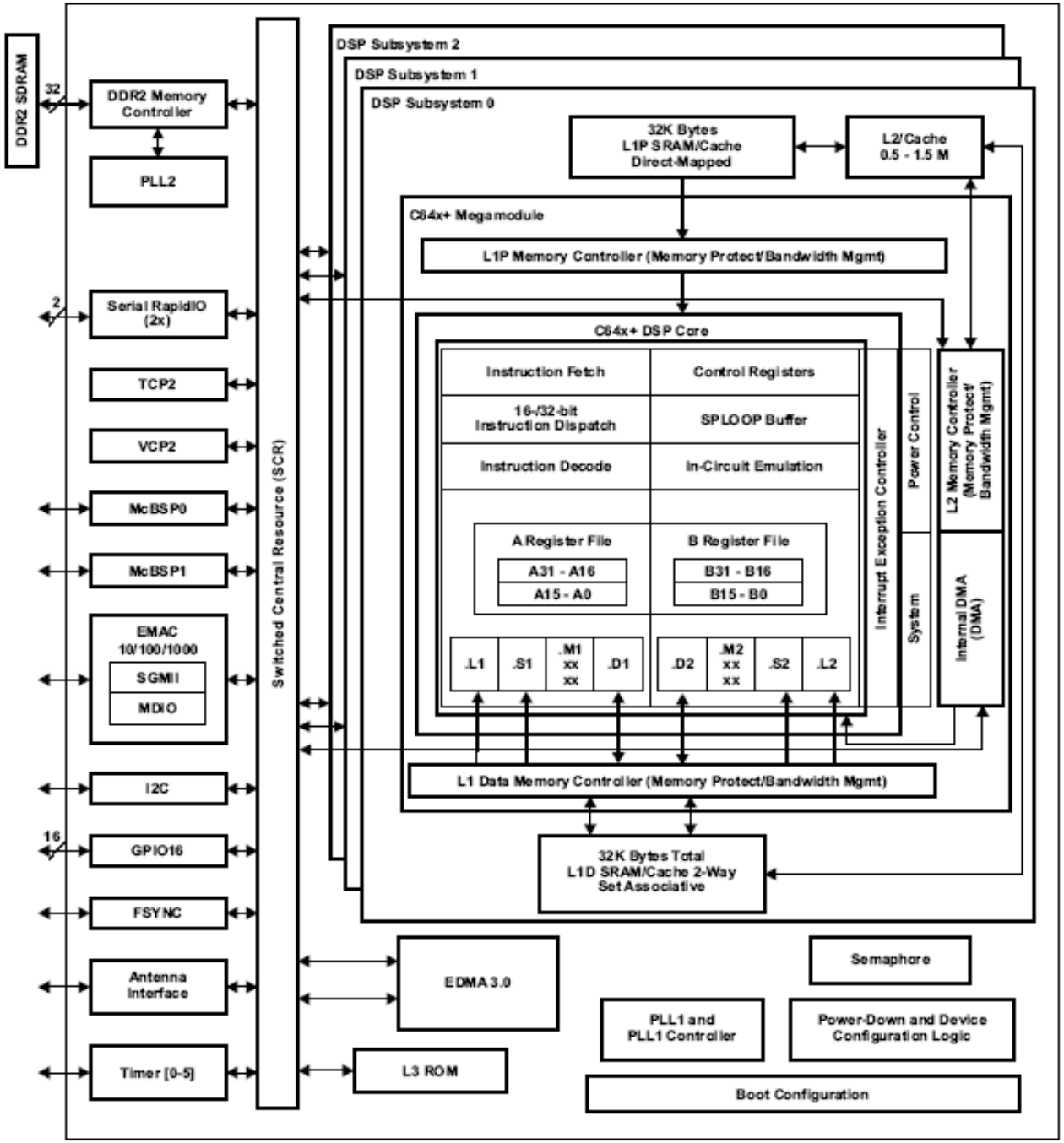


- 2001年
- 600～1000MHz
- PCI
- EMIFA/EMIFB
- L1(16KB*2)/L2(1MB)

TMS320C6455



- 2005年
- 720～1100MHz
- TMS320C64x+核
- **32×32bit乘法器**
- **1 SRIO 4X: 1.25-, 2.5-, 3.125-Gbps Link Rates**
- EMIFA+DDRII
- **L1/L2容量增大一倍**
- PCI 32bit/66MHz
- 10/100/1000 Mb/s Ethernet MAC (EMAC)



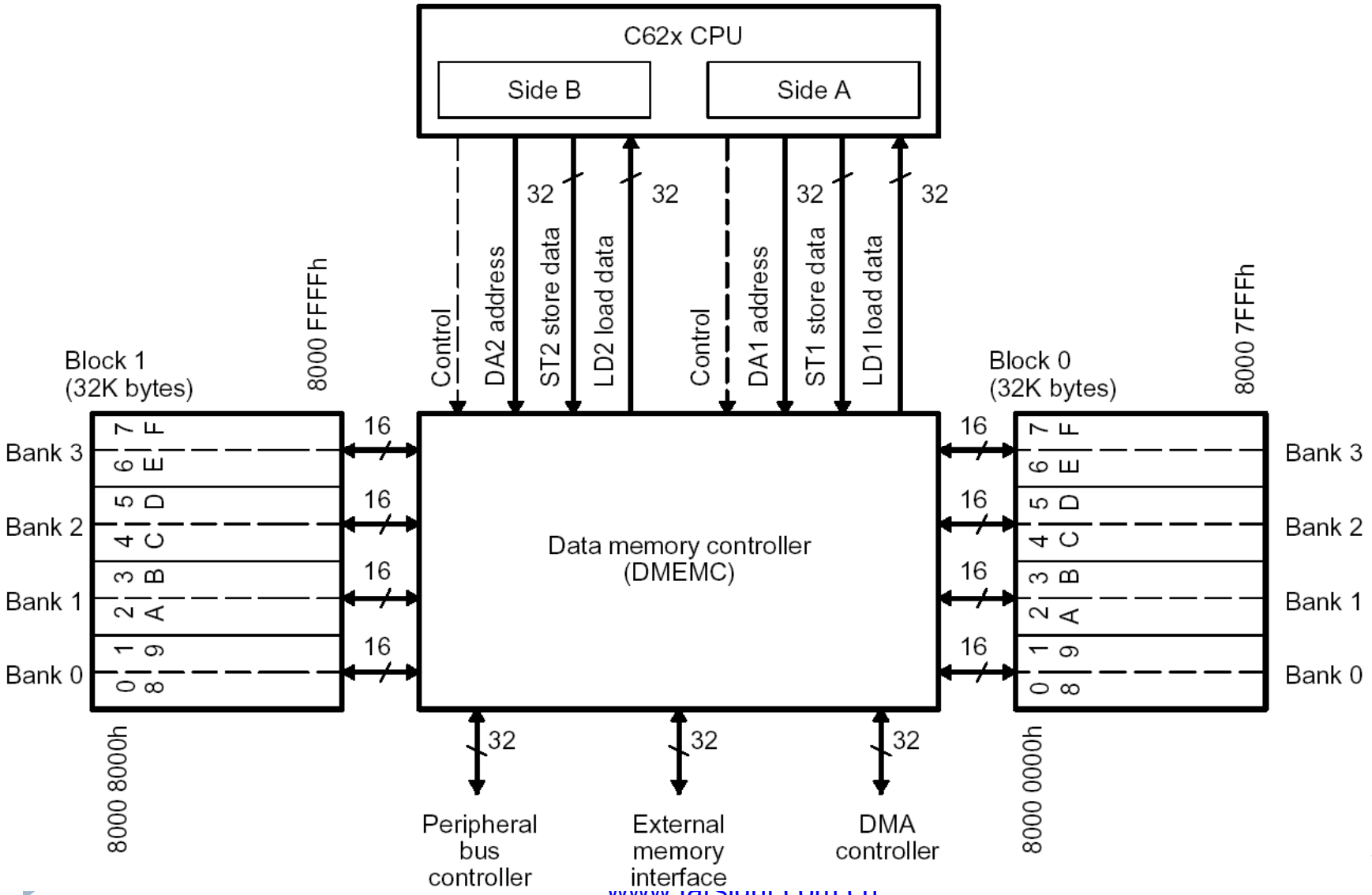
RedLogic 华清远见旗下品牌

- 2008年
- 1000MHz
- **3×TMS320C64x+核**
- 32×32bit乘法器
- SRIO 2个1X: 1.25-, 2.5-, 3.125-Gbps Link Rates
- DDRII
- EMIFA没有了
- L2－3MB，可配置
- PCI 没有
- 10/100/1**000** Mb/s Ethernet MAC (EMAC)

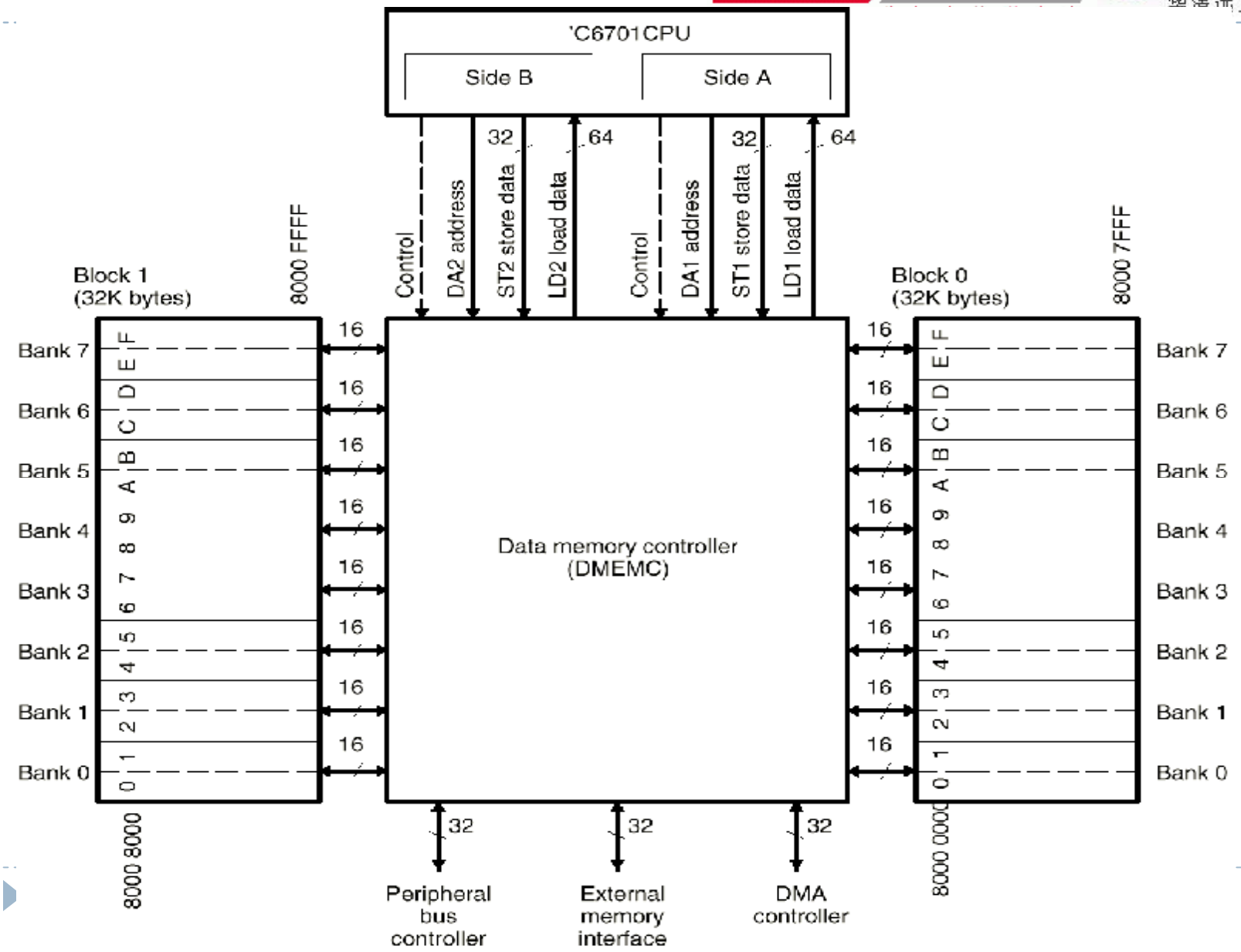
- *计划中：*

C6472：六核

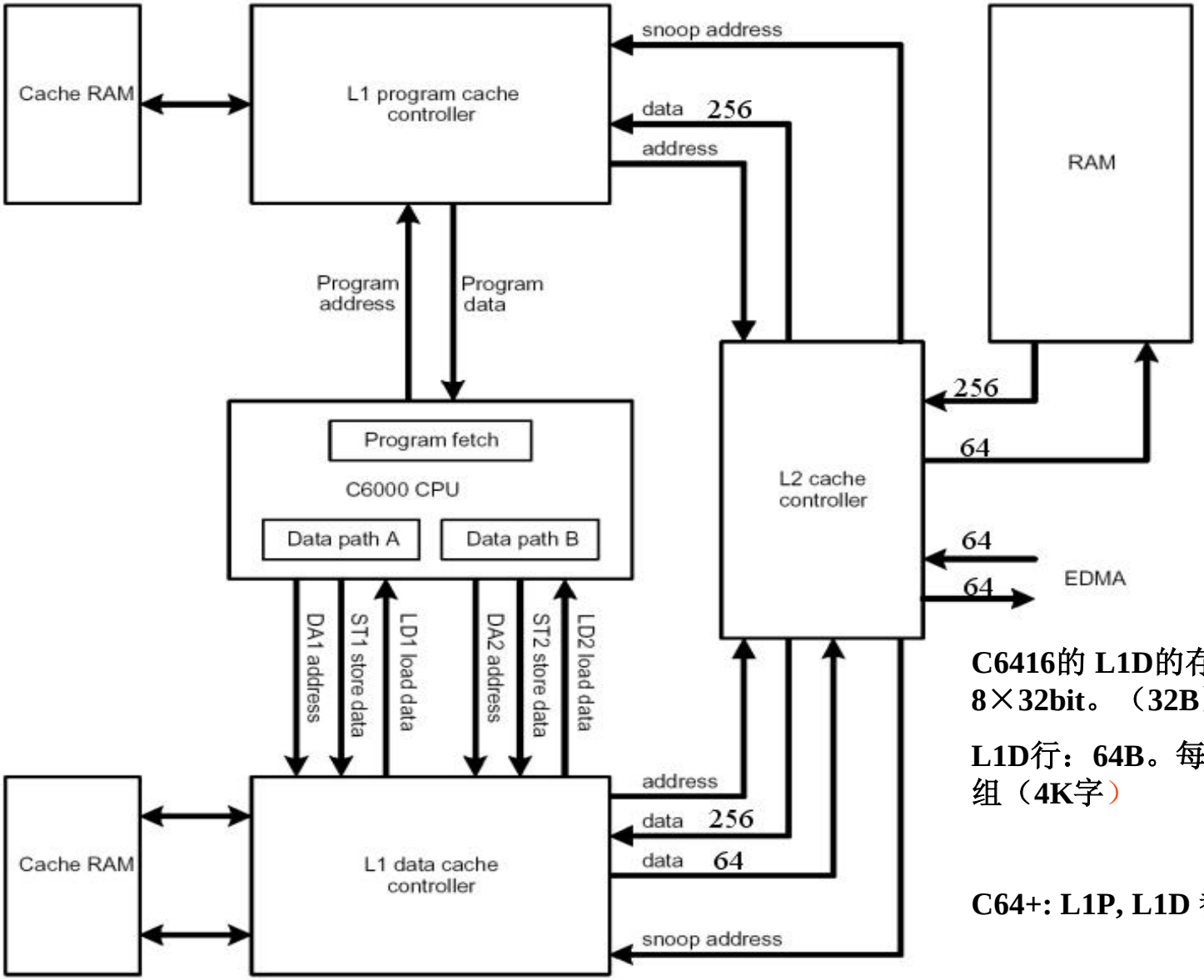
C6201/04/05 片内存储器



C6701片内存储器


'C6701 CPU
Side B
Side A
Control
DA2 address
ST2 store data
LD2 load data
Control
DA1 address
ST1 store data
LD1 load data
32
64
32
64
Block 1
(32K bytes)
8000 FFFF
Block 0
(32K bytes)
8000 7FFF
Bank 7
Bank 6
Bank 5
Bank 4
Bank 3
Bank 2
Bank 1
Bank 0
16
Data memory controller
(DMEMC)
8000 8000
8000 0000
32
Peripheral bus controller
External memory interface
DMA controller

C64x 片内2级存储器

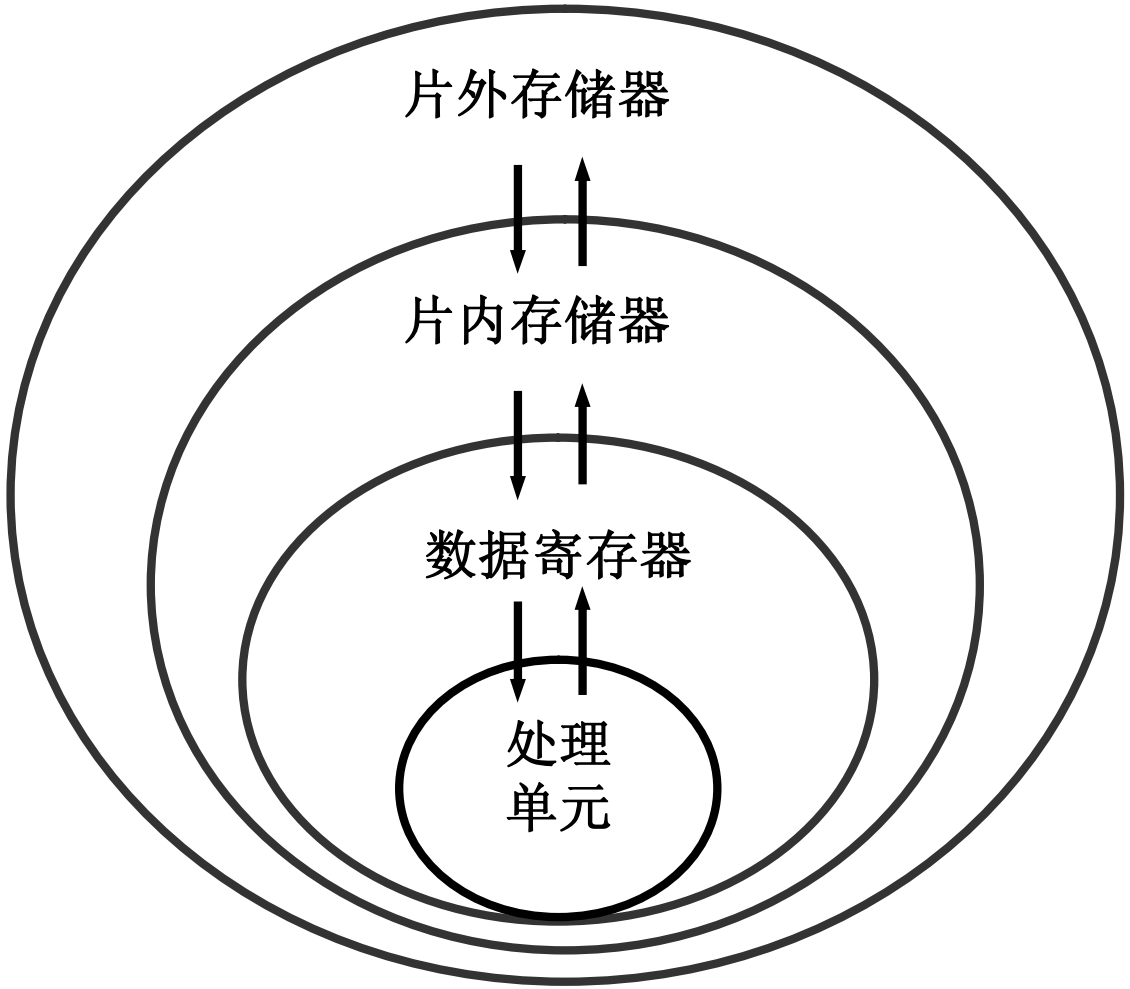


C6416的 L1D的存储体结构：8×32bit。（32B）

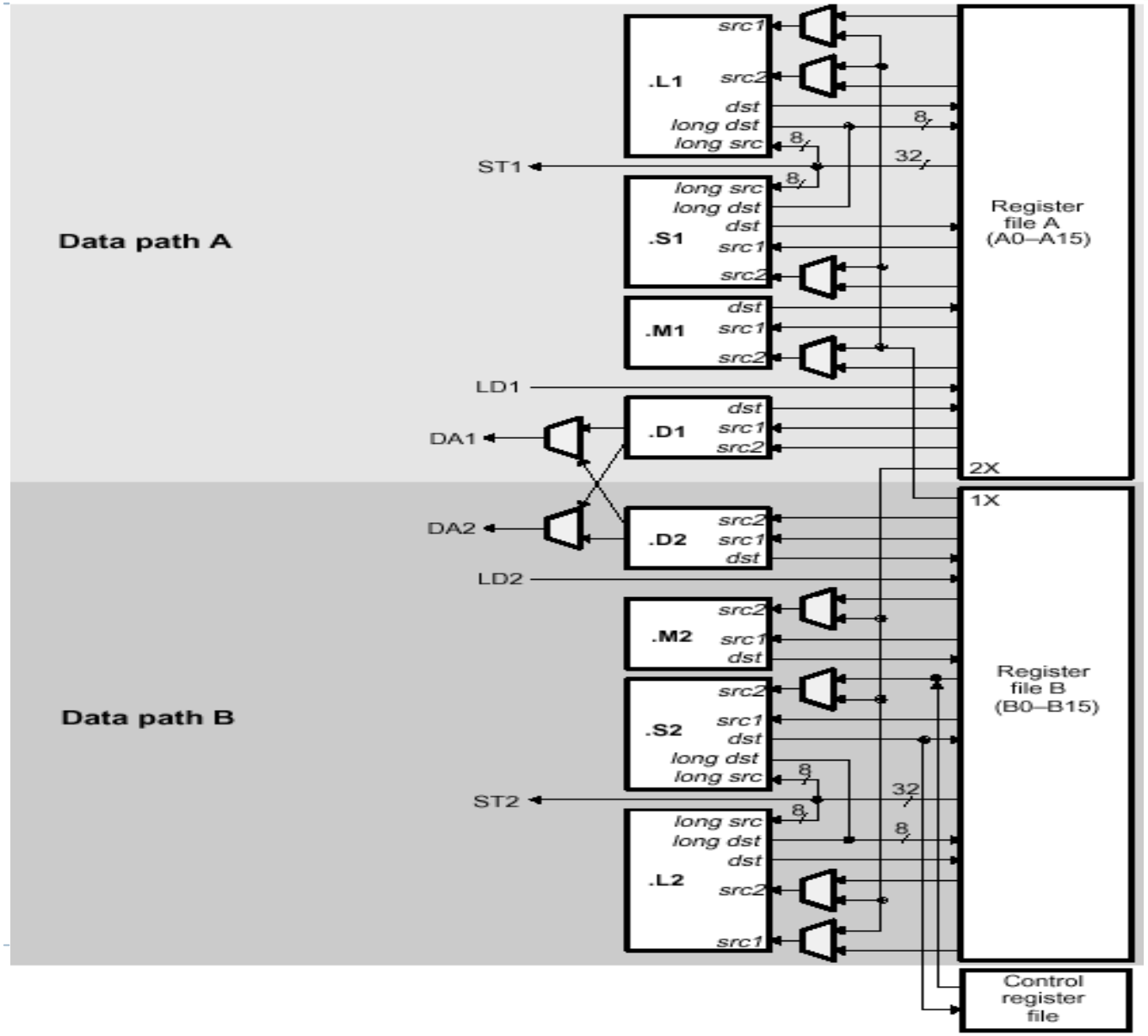
L1D行：64B。每组2行，共128组（4K字）

C64+: L1P, L1D 都是8KWord

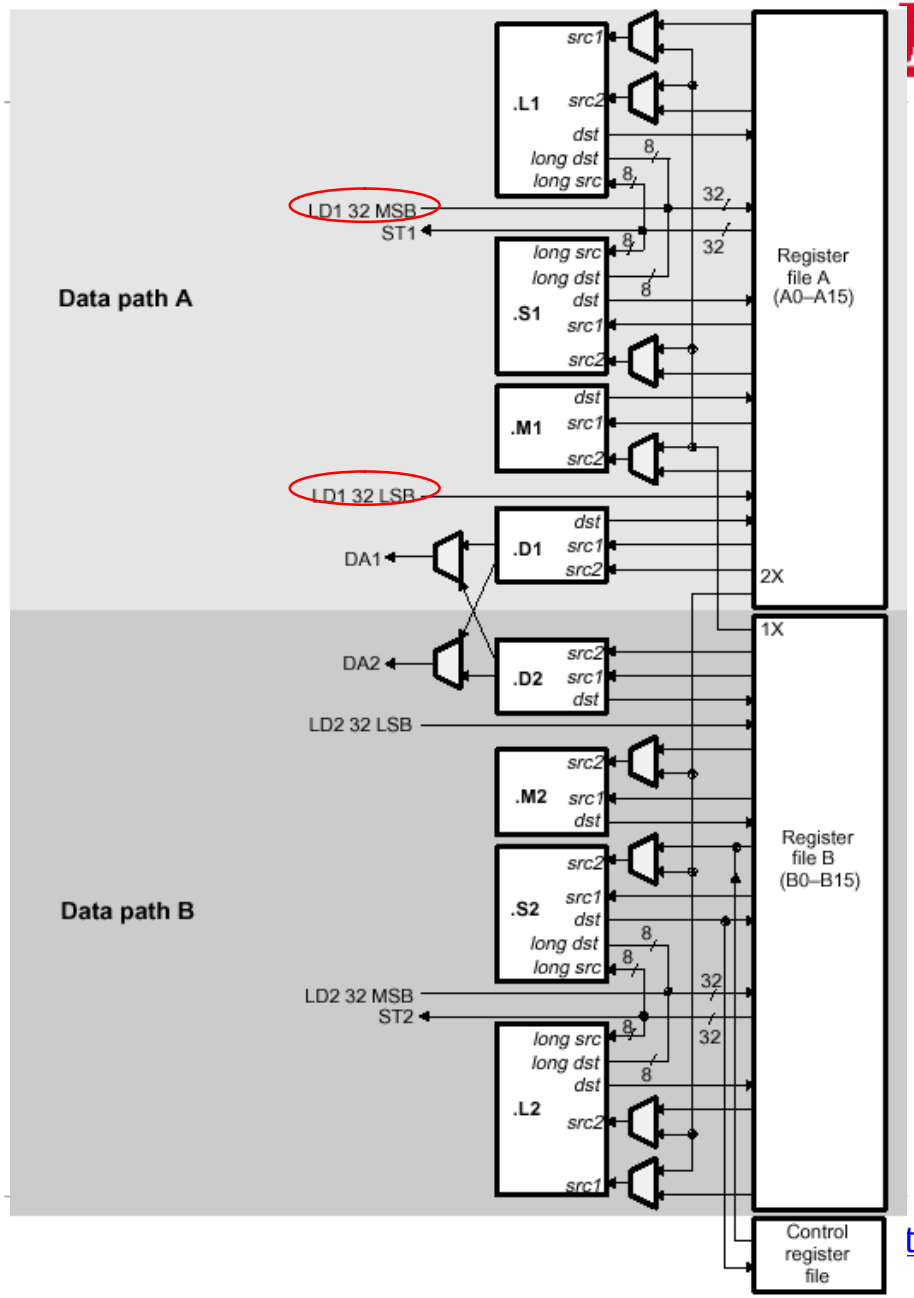
程序员角度的DSP结构：存储器的层次



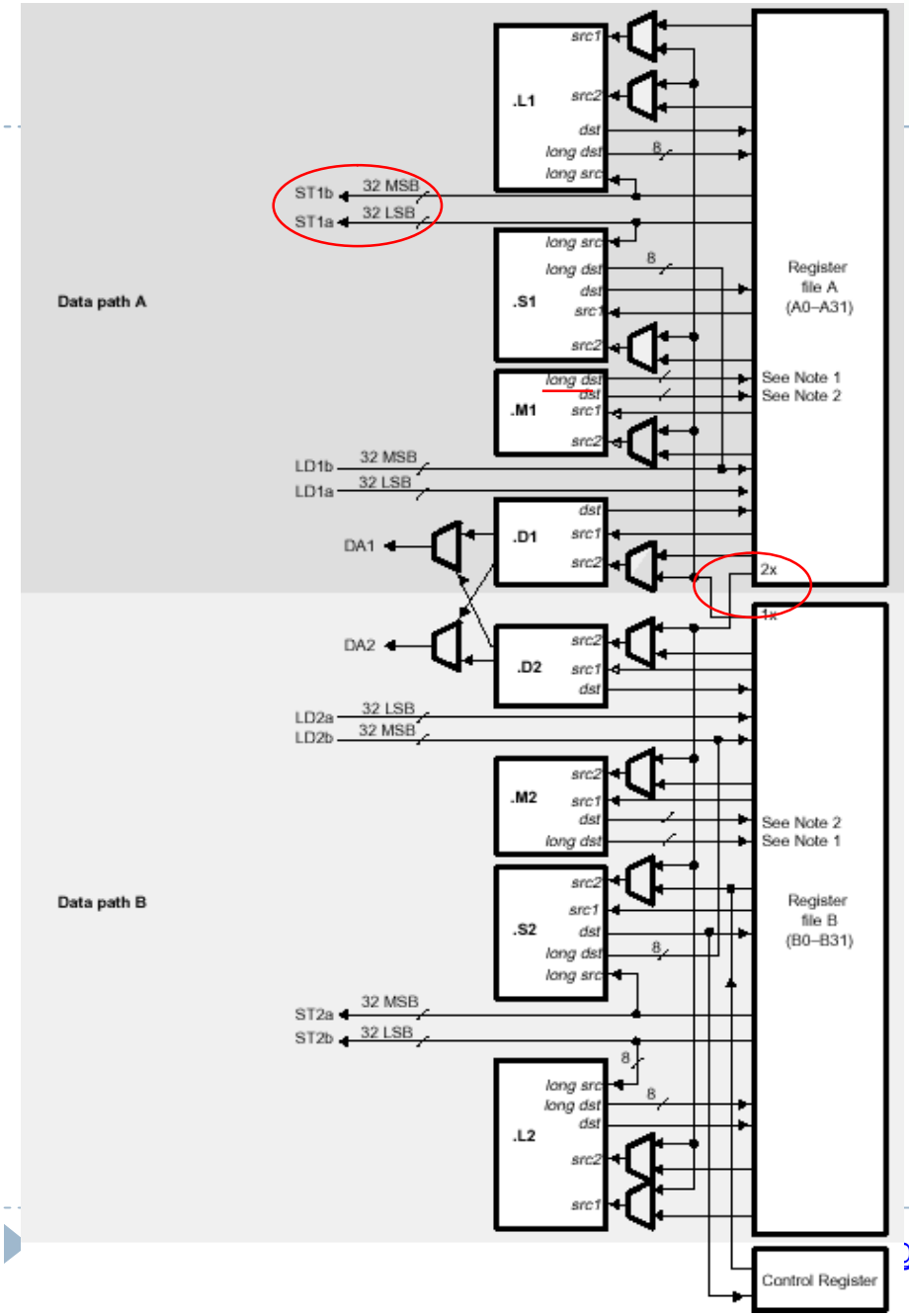
C62xx CPU Core



C67xx CPU Core



C64xx CPU Core



一条C6000的指令和其机器码

- C6000: ADD .D2 B5,B4,B4
  ADD (.D2 or.D1) src2,src1,dst1
- 00000010000101001000100001000010
- 000 0 00100 00101 00100 010000 10000 1 0
- (1) (2) (3) （4） (5) （6） （7）（8）（9）
- (1) 条件寄存器: A1,A2,B0~2; C64添加A0
- (2) z，指定条件寄存器的判断条件
- (3) dst，目的
- (4) src2，源2
- (5) src1, 源1
- (6) 操作码：设定唯一指令的码，sint，2个源和目标都为有符号整数且功能单元为D时的操作码就是010000 ；
- (7) 固定值
- (8) s，选择A边寄存器还是B边寄存器
- (9) p，是否并行

SOP （乘法累加）On C6000

3类指令：

1)运算类

2)数据搬移类

3)流程控制类

```
            MVKL    .S1     pt1, A5
            MVKH    .S1     pt1, A5

            MVKL    .S1     pt2, A6
            MVKH    .S1     pt2, A6

            MVKL    .S1     pt3, A7
            MVKH    .S1     pt3, A7
            MVKL    .S2     count, B0
            ZERO    .L1     A4
loop :      LDH     .D1     *A5++, A0
            LDH     .D1     *A6++, A1
            NOP             4
            MPY     .M1     A0, A1, A3
            NOP
            ADD     .L1     A4, A3, A4
            SUB     .S2     B0, 1, B0
  [B0]      B       .S2     loop
            NOP             5
            STW     .D1     A4, *A7
```

SOP循环核的优化

使用并行指令


```
loop:
        ldh.d1    *A8++,A2
        ldh.d1    *A9++,A3
        nop       4

        mpy.m1    A2,A3,A4
        nop
        add.l1    A4,A6,A6

        sub.l2    B0,1,B0
 [b0]   b.s1      loop
        nop       5
```

- 哪些指令可以并行？

并行指令

```
loop:
        ldh.d1    *A8++,A2
   ||   ldh.d1    *A9++,A3
        nop       4

        mpy.m1    A2,A3,A4
        nop
        add.l1    A4,A6,A6

        sub.l2    B0,1,B0
 [b0]   b.s1      loop
        nop       5
```

- 哪些指令可以并行？

(1)两条取指令并行：

放“||”在第二个ldh前

.d1改为.d2，A改为B

填充延迟间隙


```
loop:
        ldh.d1    *A8++,A2
   ||   ldh.d2    *B9++,B3
        nop       4

        mpy.m1x   A2,B3,A4
        nop
        add.l1    A4,A6,A6

        sub.l2    B0,1,B0
 [b0]   b.s1      loop
        nop       5
```

- NOP：相当于未优化
- 为了消除NOP，如何调整指令顺序？

填充延迟间隙


```
loop:
        ldh.d1    *A8++,A2
   ||   ldh.d2    *B9++,B3
        sub.l2    B0,1,B0
  [b0]  b.s1      loop
        nop       2
        mpy.m1x   A2,B3,A4
        nop
        add.l1    A4,A6,A6
```

Sub和b指令移到ldh指令后：

- LD的nop由4降为2
- B的nop被消除

填充延迟间隙优化结果

✓	**No Optimization**	
	16 cycles x 40 iterations =	**640**
✓	**Parallel Optimization**	
	15 cycles x 40 iterations =	**600**
✓	**Filling Delay Slots**	
	8 cycles x 40 iterations =	**320**
	Word Wide Optimizations	

循环代码展开举例

```
        mvk     4,B0
loop:   ldh
  ||    ldh
        mpy
        add                 ; adds 1, 2, 3, & 4
 [B0]   sub     B0,1,B0
 [B0]   b       loop
```

循环展开: 减少B的开销,但增加代码尺寸

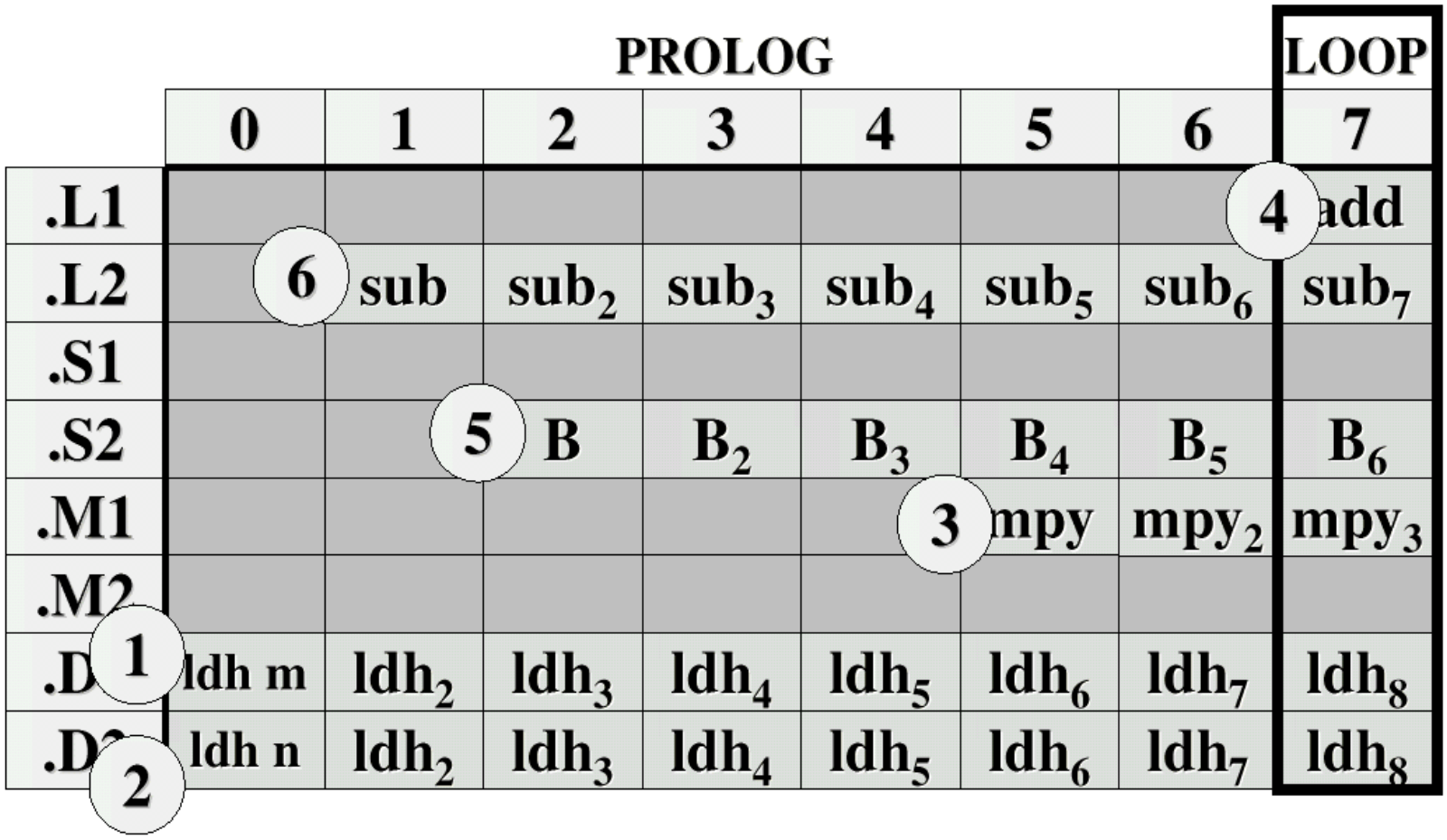

```
            mvk     2,B0
loop:       ldh
  ||        ldh
            mpy
            add                 ; adds 1 & 3

            ldh
  ||        ldh
            mpy
            add                 ; adds 2 & 4

 [B0]       sub     B0,1,B0
 [B0]       b       loop
```

- 循环次数减少一半

软件流水的SOP：编排表



Sop软件流水的汇编代码


```
c0:       ldh .D1   *A1++,A2
 ||       ldh .D2   *B1++,B2

c1:       ldh .D1   *A1++,A2
 ||       ldh .D2   *B1++,B2
 || [B0]  sub .L2   B0,1,B0

c2_3_4:   ldh .D1   *A1++,A2
 ||       ldh .D2   *B1++,B2
 || [B0]  sub .L2   B0,1,B0
 || [B0]  B   .S2   c7
             .
             .
             .
```

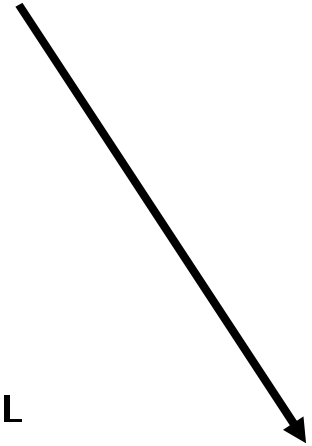
```
c5_6:     ldh   .D1   *A1++,A2
 ||       ldh   .D2   *B1++,B2
 || [B0]  sub   .L2   B0,1,B0
 || [B0]  B     .S2   c7
 ||       mpy   .M1x  A2,B2,A3
             .
             .
```

```
*** Single-Cycle Loop

c7:       ldh .D1   *A1++,A2
 ||       ldh .D2   *B1++,B2
 || [B0]  sub .L2   B0,1,B0
 || [B0]  B   .S2   c7
 ||       mpy .M1x  A2,B2,A3
 ||       add .L1   A4,A3,A4
```

```
            .global     _mpympyh
_mpympyh:   .cproc      p_x, p_a, n
             .reg       sum, sum1, sum2, x, a, prod1, prod2

              zero      sum1
              zero      sum2
              SHR       n, 1, n
loop:         ;.trip 20
              LDW       *p_x++, x
              LDW       *p_a++, a
              MPY       x, a, prod1
              MPYH      a, x, prod2
              ADD       prod1, sum1, sum1
              ADD       prod2, sum2, sum2
              SUB       n, 1, n
           [n] B        loop
              ADD       sum1, sum2, sum
            .return  sum
            .endproc
```



```
                  loop:    ; PIPED LOOP KERNEL

                              ADD     .L2     B7,B4,B4          ; |13|
                     ||       ADD     .L1     A5,A0,A0          ; |14|
                     ||       MPY     .M2X    A3,B6,B7          ; @@|11|
                     ||       MPYH    .M1X    B6,A3,A5          ; @@|12|
                     || [ B0] B       .S1     loop              ; @@@@@|16|
                     || [ B0] ADD     .S2     0xffffffff,B0,B0  ; @@@@@@@|15|
                     ||       LDW     .D1T1   *A4++,A3          ; @@@@@@@@|9|
                     ||       LDW     .D2T2   *B5++,B6          ; @@@@@@@@|10|
```

谢谢！