Technology Independent Area and Delay Estimates for Microprocessor Building Blocks

Shashank Gupta   Stephen W. Keckler   Doug Burger
Computer Architecture and Technology Laboratory
Department of Computer Sciences
Tech Report TR2000-05
The University of Texas at Austin
cart@cs.utexas.edu — www.cs.utexas.edu/users/cart

ABSTRACT

As technologies continue to shrink and sizes of chips continue to get larger, the transistor budget available for design is growing rapidly. Accurate area and delay estimates for the various structures would help in estimating the approximate area and delay estimates for blocks which could be used to make preliminary analysis of new designs. In this work, we identify some of the basic architecture building blocks and provide area and delay estimates for them. Published die photos of various processors were analyzed to make area estimates which were converted to technology independent λ^2 units considering the process parameters used for the chip fabrication. Using those estimates, a number of design configurations have been analyzed and projections about the area and delay of future designs are made.

1 Introduction

As the technologies continue to shrink and the sizes of the chips continue to get larger, the transistor budget available for design is growing rapidly. It is projected that a few years down the line we will have a billion transistors at our disposal for designing a single chip. To utilize this area budget effectively, a lot of aggressive design techniques will have to be employed.

If accurate area and delay estimates for the various structures were available, it would be possible to evaluate new ideas fairly quickly. From the straw-man architecture, rough area and delay estimates for the various blocks could be calculated and then the feasibility of the proposed design could be measured by putting those blocks together and seeing if they fall within the permissible delay and area ranges. This methodology would help to trim down the huge design space and provide some insight into what future designs might look like.

In this work, we have identified some basic architecture building blocks and come up with area and delay estimates for them. Using those estimates, a number of design configurations have been analyzed and some projections about the area and delay of future designs have been made.

2 Methodology

To estimate the area of the components, published die photos of designs were analyzed. The area of the components (marked in the die photos) was calculated and then normalized using the total chip area to get the corresponding area in square millimeters. This area was then converted to a square λ value (which is independent of technology) by using the process parameters in which the chip was fabricated.

For structures such as memories, caches and register files, dividing this square λ area by the number of bits gave the area per bit cell of the structure.

2.1 Caches

Tables 1,2 contain the areas and sizes for various instruction and data caches that were analyzed.

λ^2 Area	Area (mm2)	Size	Process (um)	λ^2/Cell	Cell (um2)	Ref#	Year
7.82e+8	12.233	64KB	0.25	1493.3	23.33	[3]	Oct'98
4.05e+8	4.9014	32KB	0.22	1545.2	18.69	[7]	Nov'98
8.84e+8	13.815	64KB	0.15	1686.4	26.35	[16]	'98
2.25e+8	3.5299	16KB	0.20	1723.5	26.93	[12]	'98
7.25e+9	222.2100	512KB	0.35	1729.9	52.97	[1]	'98
6.17e+7	0.9650	4KB	0.25	1884.7	29.44	[5]	Apr'99
1.49e+8	2.3405	8KB	0.25	2285.7	35.71	[18]	Aug'98
9.18e+7	1.4353	4KB	0.15	2803.4	43.80	[15]	'98
4.44e+8	8.7074	17KB	0.28	3190.0	62.52	[14]	'98
4.90e+8	7.6605	16KB	0.25	3740.4	58.44	[17]	Oct'98
1.13e+9	4.0852	32kB	0.12	4328.8	15.58	[13]	'98
3.45e+8	3.8091	8KB	0.21	5271.8	58.12	[11]	'98

Table 1: Data Caches

Average λ^2 cell area:

λ^2 Area	Area (mm2)	Size	Process (um)	λ^2/Cell	Cell (um2)	Ref#	Year
4.05e+8	4.9014	32KB	0.22	1545.2	18.69	[7]	Nov'98
8.23e+8	12.869	64KB	0.25	1570.9	24.54	[3]	Oct'98
8.93e+8	13.957	64KB	0.15	1703.8	26.62	[16]	'98
1.12e+8	1.7649	8KB	0.20	1723.5	26.93	[12]	'98
1.35e+8	2.1230	8KB	0.25	2073.2	32.39	[5]	Apr'99
1.60e+8	2.5077	8KB	0.25	2448.9	38.26	[18]	Aug'98
8.26e+7	1.2918	4KB	0.15	2523.0	39.42	[15]	'98
3.83e+8	7.5200	16KB	0.28	2927.2	57.37	[14]	'98
4.25e+8	6.6525	16KB	0.25	3248.3	50.75	[17]	Oct'98
1.38e+9	4.9829	32KB	0.12	5280.1	19.00	[13]	'98
3.45e+8	3.8091	8KB	0.21	5271.8	58.12	[11]	'98

Table 2: Instruction Caches

λ^2 Area	Area (mm2)	Process (um)	λ^2/Cell	Cell (um2)	Component	Ref#	Year
2.38e+8	7.3010	0.35	113.67	3.48	DRAM Bank	[19]	'98
5.45e+8	8.5307	0.25	130.16	2.03	DRAM Macro	[5]	Apr'99
5.97e+7	1.8292	0.35	810.12	24.81	SRAM	[9]	Dec'98
8.24e+6	0.2524	0.35	1006.20	30.81	I/O RAM	[9]	Dec'98
1.23e+8	1.9300	0.25	1884.70	29.44	SPRAM	[5]	Apr'99
7.07e+7	1.1052	0.15	2.23e+4	348.86	CAM (SLB)	[16]	'98
6.06e+7	0.9473	0.15	4.25e+4	665.24	CAM (BAT)	[16]	'98

Table 3: Other Memory elements

D-Cache = 2640.2
I-Cache = 2519.9
Cache = 2582.7

These numbers include the cell area together with the overheads incurred like routing and tags for the cache lines.

2.2 Memory

Areas and sizes of the memory blocks are contained in Table 3.

Average λ^2 cell area:

DRAM = 173.08
SRAM = 1359.79
CAM = 2.25e+4

2.3 Register File

The values used for the calculations below are reported in Table 4.

λ^2 Area	Area (mm2)	Size	Process (um)	λ^2/Cell	Cell (um2)	Ports	Ref#	Year
2.86e+7	0.4465	32x64b	0.25	1.39e+4	218.0	3R, 2W	[15]	'98
8.30e+7	1.2969	32x128b	0.25	2.03e+4	316.6	-	[5]	Apr'99
1.18e+8	1.8473	64x68b	0.25	2.72e+4	424.5	4R, 2W	[16]	'98
7.09e+7	1.3906	64x36b	0.28	3.08e+4	603.6	3R, 2W	[14]	'98
9.83e+6	0.3011	4x7x8b	0.35	4.39e+4	1344.2	-	[9]	Dec'98
3.31e+8	5.1820	2x64x32b	0.25	8.10e+4	1265.1	-	[6]	Dec'98
1.23e+8	1.3603	32x32b	0.21	1.20e+5	1328.4	-	[11]	'98
1.57e+8	1.7413	32x32b	0.21	1.54e+5	1700.9	-	[11]	'98

Table 4: Register File

λ^2 Area	Area (mm2)	Process (um)	Size	λ^2/Bit	Composition	Ref#	Year
6.01e+7	1.8424	0.35	32b	0.94e+6	1MAC, 1ALU	[9]	Dec'98
4.25e+7	0.8343	0.28	36b	1.18e+6	ALU/Shifter	[14]	'98
2.20e+8	3.4411	0.25	64b	1.72e+6	2 ALU	[18]	Aug'98
1.34e+8	2.0965	0.25	64b	2.09e+6	IDP	[17]	Oct'98
2.16e+8	2.6141	0.22	32b	2.25e+6	2ALU,RF,SU	[7]	Nov'98
1.72e+8	2.7020	0.25	64b	2.69e+6	ALU/Shifter	[5]	Apr'99
1.72e+8	1.9045	0.21	16b	2.69e+6	4 ALU	[11]	'98
2.06e+8	3.2226	0.25	32b	3.22e+6	2 AlU, RF?	[12]	'98
3.39e+8	5.3115	0.25	32b	3.53e+6	3 IEU ?	[4]	Mar'99
7.27e+8	11.359	0.25	32b	3.78e+6	3IEU, 3AGU	[3]	Oct'98

Table 5: Integer ALU

Average λ^2 cell area:
including all 7 values = 6.13e+4
excluding the 2 e+5 and 1.39e+4 values = 4.06e+4 *(used)*

2.4 Integer ALU

Integer ALU sizes are reported in Table 5.

Average λ^2 area for Integer ALU per bit = 2.41e+6

Average λ^2 area for a 64 bit Integer ALU = 1.54e+8

2.5 Integer Multiplier

Area and delay values for various integer multipliers are reported in Table 6.
Average λ^2 area for Integer Multiplier per bit:
over all 8 values = 2.47e+6
over all but 6.93e+4 = 1.84e+6 *(used)*
Average λ^2 area of a 64b multiplier = 1.18e+8

In the values presented, the tradeoff between area and delay of the component can be clearly seen for the cases where a delay value is available.

λ^2 Area	Area (mm2)	Process (um)	Size	λ^2/Bit	Del (ns)	FO4	Comp.	Ref#	Year
8.13e+7	1.2700	0.25	54b	1.51e+6	4.1	45.5	-	[10]	'97
1.07e+8	2.0915	0.28	64b	1.67e+6	6.0	59.5	-	[2]	'98
1.18e+8	1.8473	0.25	64b	1.84e+6	-	-	-	[16]	'98
1.21e+8	1.8914	0.25	64b	1.89e+6	-	-	Mac	[5]	Apr'99
7.09e+7	1.3906	0.28	8b	2.21e+6	-	-	4 Mac	[14]	'98
2.02e+8	3.1500	0.25	54b	3.74e+6	2.7	30.0	-	[8]	'98
4.44e+8	4.8974	0.21	16b	6.93e+6	-	-	4 Mac	[11]	'98

Table 6: Integer Multiplier

λ^2 Area	Area (mm2)	Process (um)	Size	λ^2/Bit	Composition	Ref#	Year
1.35e+8	2.1179	0.25	64b	2.10e+6	Mul, Add	[12]	'98
1.71e+8	2.6783	0.25	64b	2.67e+6	FPU, RF?	[17]	Oct'98
1.11e+8	1.7370	0.25	32b	3.47e+6	FMAC	[5]	Apr'99
1.41e+8	2.7812	0.28	32b	4.41e+6	FPU	[14]	'98
3.37e+8	4.0845	0.22	64b	5.27e+6	FPU, FP RF	[7]	Nov'98
6.01e+8	9.3923	0.25	64b	9.39e+6	FPU	[6]	Dec'98
1.05e+9	3.8128	0.12	64b	1.64e+7	FPU, RF?	[13]	'98
1.47e+9	23.037	0.25	64b	2.29e+7	FPU, Fstore, RF	[3]	Oct'98

Table 7: Floating Point Unit

2.6 Floating Point Unit

Usually, the FPU was reported as a single unit which included the ALU, multiplier, divider and in a lot of cases the register file. Where additional information was available, it is reported.

Sizes for complete Floating Point Units are report in Table 7. Table 8 contains the sizes of individual components in the floating point unit.

Average λ^2 area for FPU per bit:

over all 8 values	= 8.32e+6
over all but the two e+7 values	= 4.55e+6 *(used)*

Average λ^2 area of a 64b FPU = 2.91e+8

3 Area estimation for sample cores

In this section, the area estimations have been done for various core configurations over different technologies. The number of such cores that will fit on 70% of a chip of size 20mm x 20mm are also included with each estimation.

Table 9 shows the λ^2 area in different technologies for the target chip. Tables 10–17 show the area estimations for various configurations.

The λ^2 areas for different components calculated in the previous section have been used in these calculations. These are:

ALU per bit = 2.41e+6

λ^2 Area	Area (mm2)	Process (um)	Size	λ^2/Bit	Composition	Ref#	Year
2.11e+8	3.3049	0.25	32b	1.65e+6	4FADD(SIMD)	[4]	Mar'99
2.44e+8	2.6935	0.21	32b	3.81e+6	2 FADD	[11]	'98
6.09e+8	7.3703	0.22	32b	4.76e+6	4FMAC(Vec)	[7]	Nov'98
7.10e+8	7.8358	0.21	32b	1.11e+7	2 FMUL	[11]	'98
6.02e+7	0.9413	0.25	64b	9.41e+5	FDIV	[12]	'98
6.17e+7	0.9650	0.25	32b	1.93e+6	FDIV	[5]	Apr'99
2.41e+8	3.7771	0.25	-	-	Packed FPU	[4]	Mar'99

Table 8: Individual Floating Point Arithmetic Units

	0.250u	0.180u	0.130u	0.070u	0.05u	0.035u
Full chip	2.56e+10	4.93e+10	9.46e+10	3.26e+11	6.40e+11	1.30e+12
70% of the chip	1.79e+10	3.45e+10	6.62e+10	2.28e+11	4.48e+11	9.14e+11

Table 9: λ^2 area for a chip of size $400mm^2$

Multiplier per bit	= 1.84e+6
FPU per bit	= 4.55e+6
Register per bit	= 4.06e+4
ICache cell	= 2519
DCache cell	= 2640
SRAM cell	= 1360

4 Delay estimation for various design points

The number of cycles required to traverse the side of a core are calculated for three frequencies: SIA road-map projected frequency for the technology, 16 FO4 delay clock frequency and 8 FO4 delay clock frequency. The values for these frequencies used are reported in Table 18.

Component	Size	Num	λ^2	0.250u	0.180u	0.130u	0.070u	0.050u	0.035u
ALU	64	1	1.54e+8	2.406	1.247	0.650	0.188	0.096	0.047
Total Core Area			1.54e+8	2.406	1.247	0.65	0.188	0.096	0.047
No.Cores in 0.7 Chip				116	224	430	1489	2916	5957

Table 10: Sample Core 1

Component	Size	Num	λ^2	0.250u	0.180u	0.130u	0.070u	0.050u	0.035u
ALU	64	1	1.54e+8	2.406	1.247	0.650	0.188	0.096	0.047
RegBit	64	8	2.07e+7	0.323	0.167	0.087	0.025	0.012	0.006
Total Core Area			1.74e+8	2.729	1.414	0.737	0.213	0.108	0.053
No.Cores in 0.7 Chip				102	198	379	1314	2592	5283

Table 11: Sample Core 2

Component	Size	Num	λ^2	0.250u	0.180u	0.130u	0.070u	0.050u	0.035u
ALU	64	1	1.54e+8	2.406	1.247	0.650	0.188	0.096	0.047
Multiplier	64	1	1.17e+8	1.828	0.947	0.494	0.143	0.073	0.035
Total Core Area			2.71e+8	4.234	2.194	1.144	0.331	0.169	0.082
No.Cores in 0.7 Chip				66	127	244	845	1656	3414

Table 12: Sample Core 3

Component	Size	Num	λ^2	0.250u	0.180u	0.130u	0.070u	0.050u	0.035u
ALU	64	1	1.54e+8	2.406	1.247	0.650	0.188	0.096	0.047
Multiplier	64	1	1.17e+8	1.828	0.947	0.494	0.143	0.073	0.035
RegBit	64	8	2.07e+7	0.323	0.167	0.087	0.025	0.012	0.006
Total Core Area			2.91e+8	4.557	2.361	1.231	0.356	0.181	0.088
No.Cores in 0.7 Chip				61	118	227	786	1546	3181

Table 13: Sample Core 4

Component	Size	Num	λ^2	0.250u	0.180u	0.130u	0.070u	0.050u	0.035u
ALU	64	1	1.54e+8	2.406	1.247	0.650	0.188	0.096	0.047
Multiplier	64	1	1.17e+8	1.828	0.947	0.494	0.143	0.073	0.035
FPU	64	1	2.91e+8	4.546	2.357	1.229	0.356	0.181	0.089
RegBit	64	8	2.07e+7	0.323	0.167	0.087	0.025	0.012	0.006
Total Core Area			5.82e+8	9.103	4.718	2.46	0.712	0.362	0.177
No.Cores in 0.7 Chip				30	59	113	393	773	1581

Table 14: Sample Core 5

Component	Size	Num	λ^2	0.250u	0.180u	0.130u	0.070u	0.050u	0.035u
ALU	64	1	1.54e+8	2.406	1.247	0.650	0.188	0.096	0.047
Multiplier	64	1	1.17e+8	1.828	0.947	0.494	0.143	0.073	0.035
FPU	64	1	2.91e+8	4.546	2.357	1.229	0.356	0.181	0.089
RegBit	64	8	2.07e+7	0.323	0.167	0.087	0.025	0.012	0.006
RAM	8	256	2.78e+6	0.043	0.022	0.011	0.003	0.001	0.000
Total Core Area			5.85e+8	9.146	4.74	2.471	0.715	0.363	0.177
No.Cores in 0.7 Chip				30	59	113	391	771	1581

Table 15: Sample Core 6

Component	Size	Num	λ^2	0.250u	0.180u	0.130u	0.070u	0.050u	0.035u
ALU	64	4	6.16e+8	9.625	4.989	2.602	0.754	0.385	0.188
Multiplier	64	1	1.17e+8	1.828	0.947	0.494	0.143	0.073	0.035
FPU	64	1	2.91e+8	4.546	2.357	1.229	0.356	0.181	0.089
RegBit	64	160	4.15e+8	6.484	3.361	1.753	0.508	0.259	0.127
RegBit	64	72	1.87e+8	2.921	1.514	0.790	0.229	0.116	0.057
Total Core Area			1.62e+9	25.404	13.168	6.868	1.99	1.014	0.496
No.Cores in 0.7 Chip				11	21	40	140	276	564

Table 16: Sample Core 7

Component	Size	Num	λ^2	0.250u	0.180u	0.130u	0.070u	0.050u	0.035u
ALU	64	4	6.16e+8	9.625	4.989	2.602	0.754	0.385	0.188
Multiplier	64	1	1.17e+8	1.828	0.947	0.494	0.143	0.073	0.035
FPU	64	1	2.91e+8	4.546	2.357	1.229	0.356	0.181	0.089
RegBit	64	160	4.15e+8	6.484	3.361	1.753	0.508	0.259	0.127
RegBit	64	72	1.87e+8	2.921	1.514	0.790	0.229	0.116	0.057
DCacheCell	8	65536	1.38e+9	21.562	11.178	5.830	1.690	0.862	0.422
Total Core Area			3.00e+9	46.966	24.346	12.698	3.68	1.876	0.918
No.Cores in 0.7 Chip				5	11	22	76	149	305

Table 17: Sample Core 8

Process (um)	SIA (GHz)	8FO4 (GHz)	16FO4 (GHz)
0.250	0.75	1.35	0.69
0.180	1.25	1.93	0.97
0.130	2.10	2.67	1.34
0.070	6.00	4.96	2.48
0.050	10.00	6.94	3.47
0.035	13.50	9.92	4.96

Table 18: Clock frequencies for various technologies

No.Cores	Area (mm2)	Side (mm)	L-Area	Delay (ns)	Cycles (SIA)	Cycles (8FO4)	Cycles (16FO4)	#IALU
FullChip	400	20	2.56e+10	6.98e-10	0.52	0.94	0.48	166
1	280	16.73	1.79e+10	5.84e-10	0.44	0.79	0.4	116
2	140	11.83	8.96e+9	4.13e-10	0.31	0.56	0.28	58
4	70	8.37	4.48e+9	2.92e-10	0.22	0.39	0.2	29
8	35	5.92	2.24e+9	2.07e-10	0.15	0.28	0.14	14
16	17.5	4.18	1.12e+9	1.46e-10	0.11	0.2	0.1	7
32	8.75	2.96	5.60e+8	1.03e-10	0.08	0.14	0.07	4
64	4.37	2.09	2.79e+8	7.3e-11	0.05	0.1	0.05	2
128	2.19	1.48	1.40e+8	5.17e-11	0.04	0.07	0.03	1

Table 19: Delay estimation for a $400mm^2$ chip in 0.25um process

No.Cores	Area (mm2)	Side (mm)	L-Area	Delay (ns)	Cycles (SIA)	Cycles (8FO4)	Cycles (16FO4)	#IALU
FullChip	400	20	4.93e+10	8.76e-10	1.09	1.69	0.85	320
1	280	16.73	3.45e+10	7.33e-10	0.92	1.41	0.71	224
2	140	11.83	1.72e+10	5.18e-10	0.65	1	0.5	112
4	70	8.37	8.64e+9	3.67e-10	0.46	0.71	0.35	56
8	35	5.92	4.32e+9	2.59e-10	0.32	0.5	0.25	28
16	17.5	4.18	2.16e+9	1.83e-10	0.23	0.35	0.18	14
32	8.75	2.96	1.08e+9	1.3e-10	0.16	0.25	0.12	7
64	4.37	2.09	5.39e+8	9.16e-11	0.11	0.18	0.09	3
128	2.19	1.48	2.70e+8	6.48e-11	0.08	0.12	0.06	2
256	1.09	1.04	1.34e+8	4.56e-11	0.06	0.09	0.04	1

Table 20: Delay estimation for a $400mm^2$ chip in 0.18um process

No.Cores	Area (mm2)	Side (mm)	L-Area	Delay (ns)	Cycles (SIA)	Cycles (8FO4)	Cycles (16FO4)	#IALU
FullChip	400	20	9.46e+10	9.78e-10	2.05	2.61	1.31	614
1	280	16.73	6.62e+10	8.18e-10	1.72	2.18	1.1	430
2	140	11.83	3.31e+10	5.78e-10	1.21	1.54	0.77	215
4	70	8.37	1.65e+10	4.09e-10	0.86	1.09	0.55	107
8	35	5.92	8.28e+9	2.89e-10	0.61	0.77	0.39	54
16	17.5	4.18	4.14e+9	2.04e-10	0.43	0.54	0.27	27
32	8.75	2.96	2.07e+9	1.45e-10	0.3	0.39	0.19	13
64	4.37	2.09	1.03e+9	1.02e-10	0.21	0.27	0.14	7
128	2.19	1.48	5.18e+8	7.24e-11	0.15	0.19	0.1	3
256	1.09	1.04	2.57e+8	5.08e-11	0.11	0.13	0.07	2
512	0.55	0.74	1.30e+8	3.62e-11	0.07	0.1	0.05	1

Table 21: Delay estimation for a $400mm^2$ chip in 0.13um process

No.Cores	Area (mm2)	Side (mm)	L-Area	Delay (ns)	Cycles (SIA)	Cycles (8FO4)	Cycles (16FO4)	#IALU
FullChip	400	20	3.26e+11	9.71e-10	5.83	4.82	2.41	2117
1	280	16.73	2.28e+11	8.13e-10	4.87	4.03	2.01	1480
2	140	11.83	1.14e+11	5.75e-10	3.45	2.85	1.42	740
4	70	8.37	5.71e+10	4.06e-10	2.44	2.02	1.01	371
8	35	5.92	2.85e+10	2.87e-10	1.72	1.43	0.71	185
16	17.5	4.18	1.42e+10	2.03e-10	1.22	1.01	0.5	92
32	8.75	2.96	7.14e+9	1.44e-10	0.86	0.71	0.36	46
64	4.37	2.09	3.56e+9	1.01e-10	0.61	0.5	0.25	23
128	2.19	1.48	1.78e+9	7.19e-11	0.43	0.36	0.18	11
256	1.09	1.04	8.89e+8	5.05e-11	0.3	0.25	0.12	6
512	0.55	0.74	4.48e+8	3.59e-11	0.21	0.18	0.09	3
1024	0.27	0.52	2.20e+8	2.52e-11	0.15	0.12	0.06	1

Table 22: Delay estimation for a $400mm^2$ chip in 0.07um process

No.Cores	Area (mm2)	Side (mm)	λ^2 Area	Delay (ns)	Cycles (SIA)	Cycles (8FO4)	Cycles (16FO4)	#IALU
FullChip	400	20	6.40e+11	1.1e-09	11.02	7.65	3.82	4156
1	280	16.73	4.48e+11	9.22e-10	9.22	6.4	3.2	2909
2	140	11.83	2.24e+11	6.52e-10	6.52	4.52	2.26	1454
4	70	8.37	1.12e+11	4.61e-10	4.61	3.2	1.6	727
8	35	5.92	5.60e+10	3.26e-10	3.26	2.26	1.13	364
16	17.5	4.18	2.80e+10	2.3e-10	2.3	1.6	0.8	182
32	8.75	2.96	1.40e+10	1.63e-10	1.63	1.13	0.57	91
64	4.37	2.09	6.99e+9	1.15e-10	1.15	0.8	0.4	45
128	2.19	1.48	3.50e+9	8.15e-11	0.81	0.57	0.28	23
256	1.09	1.04	1.74e+9	5.73e-11	0.57	0.4	0.2	11
512	0.55	0.74	8.80e+8	4.08e-11	0.41	0.28	0.14	6
1024	0.27	0.52	4.32e+8	2.86e-11	0.29	0.2	0.1	3
2048	0.14	0.37	2.24e+8	2.04e-11	0.2	0.14	0.07	1

Table 23: Delay estimation for a $400mm^2$ chip in 0.05um process

No.Cores	Area (mm2)	Side (mm)	λ^2 Area	Delay (ns)	Cycles (SIA)	Cycles (8FO4)	Cycles (16FO4)	#IALU
FullChip	400	20	1.30e+12	1.58e-09	21.3	15.65	7.82	8441
1	280	16.73	9.14e+11	1.32e-09	17.82	13.09	6.55	5935
2	140	11.83	4.57e+11	9.33e-10	12.6	9.26	4.63	2967
4	70	8.37	2.28e+11	6.6e-10	8.91	6.55	3.27	1480
8	35	5.92	1.14e+11	4.67e-10	6.3	4.63	2.32	740
16	17.5	4.18	5.71e+10	3.3e-10	4.45	3.27	1.63	371
32	8.75	2.96	2.85e+10	2.33e-10	3.15	2.32	1.16	185
64	4.37	2.09	1.42e+10	1.65e-10	2.22	1.63	0.82	92
128	2.19	1.48	7.15e+9	1.17e-10	1.58	1.16	0.58	46
256	1.09	1.04	3.55e+9	8.2e-11	1.11	0.81	0.41	23
512	0.55	0.74	1.79e+9	5.84e-11	0.79	0.58	0.29	12
1024	0.27	0.52	8.81e+8	4.1e-11	0.55	0.41	0.2	6
2048	0.14	0.37	4.57e+8	2.92e-11	0.39	0.29	0.14	3
4096	0.07	0.26	2.28e+8	2.05e-11	0.28	0.20	0.10	1

Table 24: Delay estimation for a 400mm^2 chip in 0.035um process

A Area Estimation Tool

A tool has been developed to report area estimates for different configurations of the core. It takes in a configuration and (optionally) λ^2 areas of the components and generates area estimates for the core in various technologies.The estimates derived above have been included as default parameters in the tool.Additionally, the tool also takes in as parameters the target chip area (default: $400mm^2$) and a value for the fraction of the chip that is devoted to the cores (default: 70%) and reports the number of cores that will fit on it.

The default values in the tool can be overwritten by the values specified in the area and configuration files. The values specified on the command line get the highest precedence and overwrite any other values.

A.1 Command line arguments

The tools can take the following command line arguments.

- -afile <area file>

  This option can be used to specify a area file. The file contains λ^2 areas of components that can be used to build the core. This file can define new components and also overwrite the default values for the components already defined.

  Example:

  -afile areas.txt

  A sample area file is included in the next section.

- -cfile <configuration file>

  This option can be used to specify a configuration for the core. The components that are used in the configuration should have been already defined.

  Example:

  -cfile config.txt

  A sample configuration file is included later.

- -a <compname> <lambda square area of component>

  This option is used to define a new component. It can also be used to overwrite the default area of a predefined component or the area of a component defined in the area file.

  Example:

  -a TRACECACHE 2.5e+4

- -set <setname> <compname>

  This option can be used to give a new name to a particular configuration of a component.

  Example:

  If the core can have two register sets, they can be defined as

  -set r1 REG

  -set r2 REG

  All the components that are defined are also get included in the list of set names.

- -s <setname> <size of component for the set>

  This option is used to specify the size of the component in the set. This value overwrites the default values or values specified in the configuration file.

  Example:

  -s r1 32

  -s ALU 64

- -n <setname> <number of units>

  This option is used to specify the number of components of the set in the configuration. This value overwrites the default values or values specified in the configuration file.

  Example:

  -n r1 512

  -n ALU 4

- -chip < mm^2 area of the chip>

  This option is used to specify the total area of the target chip in square millimeters.

  Example:

  -chip 400

- -chipfrac <percentage of chip used for cores>

  This option is used to specify the fraction of the chip area that is used for cores.

  Example:

  -chipfrac 70

A.2 Default Parameters

The tool contains the following set of default values.

Component	λ^2 Area	Sizes	Number
ALU	2.41e+6	64	1
MULT	1.84e+6	64	0
FPU	4.55e+6	64	0
REG	4.06e+4	1	0
ICACHE	2519	1	0
DCACHE	2640	1	0
RAM	1360	1	0

The default chip size is $400mm^2$ and 70 percent of this area is used for cores.

A.3 Sample area file

Following is a sample of the area file that can be given. The first column contains the component name and the second one contains the λ^2 area for it. Anything following / / is a comment.

```
// BEGIN OF AREA FILE

// Component names and their Lambda Square Areas

ALU        2.50e+6          // Area for a 1bit ALU
MULT       3.00e+6          // Area for a 1bit Multiplier
FPU        7.15e+6          // Area for a 1bit FPU
ICACHE     1300             // Area for an ICache Cell
DCACHE     1500             // Area for a DCache Cell
REG        3.50e+4          // Area for a 1bit register
DRAM       200              // Area for a SRAM cell


// END OF AREA FILE
```

A.4 Sample configuration file

Following is a sample of the configuration file that can be given. Anything following / / is a comment.

```
// BEGIN OF CONFIGURATION FILE

// SetName Component  Size    Number

       A       ALU       64      4         // 4          - 64b ALUs
       M       MULT      64      1         // 1          - 64b Multiplier
       F       FPU       64      1         // 1          - 64b FPU
       RF1     REG       64    160         // 160         - 64b Registers
       RF2     REG       64     72         // 72         - 64b Registers
       DC      DCACHE     8   1024         // A 64KB DCache
       DRAM    DRAM      32  65536         // 256KB DRAM

// SetName is used so that you can have multiple components of the same
// type but with different sizes. (eg. RF1,RF2)

// END OF CONFIGURATION FILE
```

A.5 Invocation Examples

This section contains some sample invocations of the tool and their results.

```
> cores.perl -help

-----------------------------------------------------------------
                      Command Line Usage
```

```
---------------------------------------------------------------
cores.perl -afile <area file>
                -cfile <configuration file>
                -a <compname>  <lambda square area of component>
                -s <setname> <size of component for the set>
                -n <setname> <number of units>
                -set <setname> <compname>
                -chip <mm2 area of the chip>
                -chipfrac <percentage of chip used for cores>

> cores.perl

Chip Area =  400 mm2
-----------------------------------------------------------------------------
Component Size    Num  L-Area   0.250u  0.180u  0.130u  0.070u  0.050u  0.035u
-----------------------------------------------------------------------------
ALU         64      1  1.54e+8   2.406   1.247   0.650   0.188   0.096   0.047

Total Area             1.54e+8   2.406   1.247    0.65   0.188   0.096   0.047

Number of Cores ( 0.7 Chip)        116     224     430    1489    2916    5957
-----------------------------------------------------------------------------


> cores.perl -n ALU 4 -n MULT 1 -n FPU 2

Chip Area =  400 mm2
-----------------------------------------------------------------------------
Component Size    Num  L-Area   0.250u  0.180u  0.130u  0.070u  0.050u  0.035u
-----------------------------------------------------------------------------
MULT        64      1  1.17e+8   1.828   0.947   0.494   0.143   0.073   0.035
ALU         64      4  6.16e+8   9.625   4.989   2.602   0.754   0.385   0.188
FPU         64      2  5.82e+8   9.093   4.714   2.458   0.712   0.363   0.178

Total Area             1.31e+9  20.546   10.65   5.554   1.609   0.821   0.401

Number of Cores ( 0.7 Chip)         13      26      50     174     341     698
-----------------------------------------------------------------------------


> cores.perl -set r1 REG -n r1 128 -s r1 32 -set r2 REG -n r2 64 -s r2 16
  -chip 300 -chipfrac 65 -n ALU 4 -n MULT 1 -s MULT 32

Chip Area =  300 mm2
-----------------------------------------------------------------------------
Component Size    Num  L-Area   0.250u  0.180u  0.130u  0.070u  0.050u  0.035u
-----------------------------------------------------------------------------
MULT        32      1  5.88e+7   0.918   0.476   0.248   0.072   0.036   0.018
ALU         64      4  6.16e+8   9.625   4.989   2.602   0.754   0.385   0.188
REG         32    128  1.66e+8   2.593   1.344   0.701   0.203   0.103   0.050
REG         16     64  4.15e+7   0.648   0.336   0.175   0.050   0.025   0.012
```

```
Total Area              8.82e+8  13.784   7.145   3.726   1.079   0.549   0.268

Number of Cores (0.65 Chip)          14      27      52     180     355     727
-------------------------------------------------------------------------------
```

```
> cores.perl -chip 300 -chipfrac 65 -n A 3 -n M 1 -s M 32 -afile areas.txt
  -cfile conf.txt

Chip Area =  300 mm2
-------------------------------------------------------------------------------
Component Size    Num  L-Area   0.250u  0.180u  0.130u  0.070u  0.050u  0.035u
-------------------------------------------------------------------------------
ALU         64      3  4.80e+8    4.80   3.888   2.028   0.588   0.588   0.147
DRAM        32  65536  4.19e+8   6.546   3.393   1.770   0.513   0.261   0.128
FPU         64      1  4.57e+8   7.140   3.701   1.930   0.559   0.285   0.139
DCACHE       8   1024  1.22e+7   0.190   0.098   0.051   0.014   0.007   0.003
REG         64    160  3.58e+8   5.593   2.899   1.512   0.438   0.223   0.109
REG         64     72  1.61e+8   2.515   1.304   0.680   0.197   0.100   0.049
MULT        32      1  9.60e+7    9.60   0.777   0.405   0.117   0.117   0.029

Total Area              1.98e+9  36.384   16.06   8.376   2.426   1.581   0.604

Number of Cores (0.65 Chip)           5      12      23      80     123     322
-------------------------------------------------------------------------------
```

```
> cores.perl -chip 300 -chipfrac 65 -n A 3 -n M 1 -s M 32 -afile areas.txt
  -cfile conf.txt -a TLB 25000 -n TLB 512 -s TLB 32

Chip Area =  300 mm2
-------------------------------------------------------------------------------
Component Size    Num  L-Area   0.250u  0.180u  0.130u  0.070u  0.050u  0.035u
-------------------------------------------------------------------------------
ALU         64      3  4.80e+8    4.80   3.888   2.028   0.588   0.588   0.147
TLB         32    512  4.09e+8   6.390   3.312   1.728   0.501   0.255   0.125
DRAM        32  65536  4.19e+8   6.546   3.393   1.770   0.513   0.261   0.128
FPU         64      1  4.57e+8   7.140   3.701   1.930   0.559   0.285   0.139
DCACHE       8   1024  1.22e+7   0.190   0.098   0.051   0.014   0.007   0.003
REG         64    160  3.58e+8   5.593   2.899   1.512   0.438   0.223   0.109
REG         64     72  1.61e+8   2.515   1.304   0.680   0.197   0.100   0.049
MULT        32      1  9.60e+7    9.60   0.777   0.405   0.117   0.117   0.029

Total Area              2.39e+9  42.774  19.372  10.104   2.927   1.836   0.729

Number of Cores (0.65 Chip)           4      10      19      66     106     267
-------------------------------------------------------------------------------
```

A.6 Code

```
#!/lusr/bin/perl

#******************************************************************************
# Author:            Shashank Gupta
#
# Email:             shashank@ece.utexas.edu
#
# Date:              25th April, 2000
#
# Project:           Technology Independent Area and Delay Estimates for
#                    Microprocessor Building Blocks
#
# Description:       This tool takes in lambda square area values for
#                    various components, configuration of the core and
#                    chip size as input and generates area of the core
#                    together with the number of such cores which will fit
#                    in on the chip for various technologies.
#
#******************************************************************************

#-----------------------------------------------------------------------------
#                         Command line options
#-----------------------------------------------------------------------------
# -afile <area file>
# -cfile <configuration file>
# -a <compname>  <lambda square area of component>
# -s <setname> <size of component for the set >
# -n <setname> <number of units>
# -set <setname> <compname>
# -chip <mm2 area of the chip>
# -chipfrac <percentage of chip used for cores>

# The component names serve as predefined SET names.

# This script has
#   - predefined lambda square areas for some components
#   - predefined sizes for ALU, Multiplier and FPU (= 64b)
#   - a predefined configuration containing one ALU
#
# All these values can be overridden using area and configuration files or
# command line options.

# Decreasing order of precedence
#
# 1. Command line value (will overwrite both below)
# 2. Value specified in file (will overwrite default)
# 3. Default value


#-----------------------------------------------------------------------------
#     Sample Area File (Anything after // is a comment and is ignored)
#-----------------------------------------------------------------------------
```

```
#
# // BEGIN OF AREA FILE
#
# // Component names and their Lambda Square Areas
#
# ALU        2.41e+6           // Area for a 1bit ALU
# MULT       1.84e+6           // Area for a 1bit Multiplier
# FPU        4.55e+6           // Area for a 1bit FPU
# ICACHE     2519              // Area for an ICache Cell
# DCACHE     2640              // Area for a DCache Cell
# REG        4.06e+4           // Area for a 1bit register
# RAM        1360              // Area for a SRAM cell
#
# // END OF AREA FILE
#

#------------------------------------------------------------------------------
# Sample Configuration File  (Anything after // is a comment and is ignored)
#------------------------------------------------------------------------------
#
# // BEGIN OF CONFIGURATION FILE
#
# // SetName Component  Size    Number
#
#  A ALU       64      4 // 4  - 64b ALUs
#  M MULT      64      1 // 1  - 64b Multiplier
#  F FPU       64      1 // 1  - 64b FPU
#  RF1 REG       64      160 // 160 - 64b Registers
#  RF2 REG       64      72 // 72 - 64b Registers
#  DC DCACHE     8      65536         // A 64KB DCache
#
# // SetName is used so that you can have multiple components of the same
# // type but with different sizes. (eg. RF1,RF2)
#
# // END OF CONFIGURATION FILE
#
#

#------------------------------------------------------------------------------
#                            Sample Invocations
#------------------------------------------------------------------------------
#
# #- This prints the command line usage
#
# > cores.perl -help
#
# #- Print the values for the default configuration (1-64b ALU)
#
# > cores.perl
#
# #- Configuration = 4-64b ALUs, 1-64b MULT, 2-64b FPUs
#
# > cores.perl -n ALU 4 -n MULT 1 -n FPU 2
```

```
#
# #- Configuration = 4-64b ALUs, 1-32b MULT, 128-32b REGs, 64-16b REGs
# #- ChipSize = 300mm2, Fraction of chip used for cores = 65%
#
# > cores.perl -set r1 REG -n r1 128 -s r1 32 -set r2 REG -n r2 64
#   -s r2 16 -chip 300 -chipfrac 65 -n ALU 4 -n MULT 1 -s MULT 32
#
# #- Configuration from conf.txt
# #- Area from areas.txt
# #- Set A (from conf.txt) redifined to have 3 components
# #- Set M (from conf.txt) redefined t0 have 1-32b component
#
# > cores.perl -chip 300 -chipfrac 65 -n A 3 -n M 1 -s M 32 -afile
#   areas.txt -cfile conf.txt
#
# #- ChipSize = 300mm2, Fraction of chip used for cores = 65%
# #- Configuration from conf.txt
# #- Area from areas.txt
# #- Set A (from conf.txt) redefined to have 3 components
# #- Set M (from conf.txt) redefined t0 have 1-32b component
# #- Defined a new component TLB with area 25000
# #- Included 512-32b TLB in configuration
#
# > cores.perl -chip 300 -chipfrac 65 -n A 3 -n M 1 -s M 32 -afile
#   areas.txt -cfile conf.txt -a TLB 25000 -n TLB 512 -s TLB 32
#
#------------------------------------------------------------------------------


# Default Chip Parameters

$chipSize = 400;
$chipFrac = 0.7;

# Set of technologies for which calculations are made

@tech = (0.25, 0.18, 0.13, 0.07, 0.05, 0.035);

# Default Lambda Square Areas

$ALU     =   2.41e+6;          # Area for a 1bit ALU
$MULT    =   1.84e+6;          # Area for a 1bit Multiplier
$FPU     =   4.55e+6;  # Area for a 1bit FPU
$ICACHE  =   2519;             # Area for an ICache Cell
$DCACHE  =   2640;             # Area for a DCache Cell
$REG     =   4.06e+4;          # Area for a 1bit register
$RAM     =   1360;             # Area for a SRAM cell

# Default Size of components

$ALUSIZE    =   64;
$MULTSIZE   =   64;
$FPUSIZE    =   64;
```

```
$ICACHESIZE = 1;
$DCACHESIZE  = 1;
$REGSIZE = 1;
$RAMSIZE = 1;

# Default Core Configuration

$ALUNUM     =   1;
$MULTNUM   =   0;
$FPUNUM    =   0;
$ICACHENUM = 0;
$DCACHENUM  = 0;
$REGNUM = 0;
$RAMNUM = 0;

# Building default arrays for values

# Size for default SETs

%size = ("ALU",$ALUSIZE,
 "MULT",$MULTSIZE,
 "FPU",$FPUSIZE,
 "ICACHE",$ICACHESIZE,
 "DCACHE",$DCACHESIZE,
 "REG",$REGSIZE,
 "RAM",$RAMSIZE);

# Composition of default SETs

%composition = ("ALU","ALU",
"MULT","MULT",
"FPU","FPU",
"ICACHE","ICACHE",
"DCACHE","DCACHE",
"REG","REG",
"RAM","RAM");

# Default Configuration (#num for each SET)

%num = ("ALU",$ALUNUM,
 "MULT",$MULTNUM,
 "FPU",$FPUNUM,
 "ICACHE",$ICACHENUM,
 "DCACHE",$DCACHENUM,
 "REG",$REGNUM,
 "RAM",$RAMNUM);


# Area of default components

%area = ("ALU",$ALU,
 "MULT",$MULT,
 "FPU", $FPU,
```

```
 "ICACHE",$ICACHE,
 "DCACHE",$DCACHE,
 "REG",$REG,
 "RAM",$RAM);

$" = ":";
$ARGLINE = join($",@ARGV);
$ARGLINE = ":$ARGLINE:";

while (!(($ARGLINE =~ /^$/)||($ARGLINE =~ /^:\s*:$/))) {
  $matched = 0;

  if ($ARGLINE =~ /-chipfrac\s*:\s*(\w*)\s*:/) {
$ARGLINE =~ s/-chipfrac\s*:\s*(\w*)\s*:/:/;
  $matched = 1;

$chipFrac = $1/100;
  }

  if ($ARGLINE =~ /-chip\s*:\s*(\w+)\s*:/) {
$ARGLINE =~ s/-chip\s*:\s*(\w*)\s*:/:/;
  $matched = 1;

$chipSize = $1;
  }

  if ($ARGLINE =~ /-afile\s*:\s*(\S*?)\s*:/) {
$ARGLINE =~ s/-afile\s*:\s*(\S*?)\s*:/:/;

  $matched = 1;

# Read Area File

open (AREAS,$1) || die "ERROR: Area File $1 not found\n";

while (<AREAS>) {
    s/\/\/.*//;

  if (/(\S+)\s+(\S+)/) {
    $size{$1} = 1;
    $composition{$1} = $1;
    $area{$1} = $2;
    $num{$1} = ($num{$1} > 0) ? $num{$1} : 0;
  }
}
close (AREAS);

  }

  if ($ARGLINE =~ /-cfile\s*:\s*(\S+?)\s*:/) {
$ARGLINE =~ s/-cfile\s*:\s*(\S+?)\s*:/:/;

  $matched = 1;
```

```
# Read Configuration File

open (CONF,$1)|| die "ERROR: Configuration File $1 not found\n";

$num{ALU} = 0;

while (<CONF>) {
    s/\/\/.*//;
  if (/(\S+)\s+(\S+)\s+(\S+)\s+(\S+)/) {
    $size{$1} = $3;
    $composition{$1} = $2;
    $num{$1} = $4;
  }
}
  }

  if ($ARGLINE =~ /-a\s*:\s*(\w*)\s*:\s*(\w+)\s*:/) {
$ARGLINE =~ s/-a\s*:\s*(\w*)\s*:\s*(\w*)\s*:/:/;
  $matched = 1;

$comp = $1;
$compArea = $2;

$area{$comp} = $2;
$size{$comp} = ($size{$comp} > 1) ? $size{$comp}  :  1;
$composition{$comp} = ($composition{$comp} != "") ?
$composition{$comp}  :  $comp;
$num{$comp} = ($num{$comp} > 0) ? $num{$comp}  :  0;
  }

  $components = join($",keys(%area));
  $components = ":$components:";

  if ($ARGLINE =~ /-set\s*:\s*(\S*?)\s*:\s*(\S*?)\s*:/) {
$ARGLINE =~ s/-set\s*:\s*(\S*?)\s*:\s*(\S*?)\s*:/:/;
  $matched = 1;

$set = $1;
$comp = $2;

if ($components =~ /:\s*$comp\s*:/) {
  $size{$set} = 1;
  $num{$set} = 0;
  $composition{$set} = $comp;
}
else {
  print "ERROR: Component $comp does not exist\n";
  exit(0);
}
  }

  $sets = join($",keys(%size));
```

```
  $sets = ":$sets:";

  if ($ARGLINE =~ /-s\s*:\s*(\w*)\s*:\s*(\w+)\s*:/) {
$ARGLINE =~ s/-s\s*:\s*(\w*)\s*:\s*(\w*)\s*:/:/;
  $matched = 1;

$set = $1;
$setSize = $2;

if ($sets =~ /:\s*$set\s*:/) {
  $size{$set} = $setSize;
}
else {
  print "ERROR: Set $set does not exist\n";
          exit(0);
        }
  }

  if ($ARGLINE =~ /-n\s*:\s*(\w*)\s*:\s*(\w+)\s*:/) {
$ARGLINE =~ s/-n\s*:\s*(\w*)\s*:\s*(\w*)\s*:/:/;
  $matched = 1;

$set = $1;
$setNum = $2;

if ($sets =~ /:\s*$set\s*:/) {
  $num{$set} = $setNum;
}
else {
  print "ERROR: Set $set does not exist\n";
          exit(0);
        }
  }


  if ($matched == 0) {

        print "-----------------------------------------------------------------\n";
print "                      Command Line Usage\n";
        print "-----------------------------------------------------------------\n";
print "cores.perl -afile <area file> \n";
print " \t\t-cfile <configuration file> \n";
print " \t\t-a <compname>  <lambda square area of component> \n";
print " \t\t-s <setname> <size of component for the set> \n";
print " \t\t-n <setname> <number of units> \n";
print " \t\t-set <setname> <compname> \n";
print " \t\t-chip <mm2 area of the chip>\n";
print " \t\t-chipfrac <percentage of chip used for cores>\n\n";
        exit(0);
  }

  while ($ARGLINE =~ /:\s*:/) {
$ARGLINE =~ s/:\s*:/:/;
```

```
  }
  if ($ARGLINE =~ /^\s*:\s*$/) {
$ARGLINE = "";
  }

}


$~ = TopLine;
write;

@sets = keys(%size);

FORLOOP:
for ($i=0; $i<@sets; $i++) {
$compname = $composition{$sets[$i]};
$compsize = $size{$sets[$i]};
$compnum = $num{$sets[$i]};

$carea[0] = $area{$compname} * $compsize * $compnum;

if ($carea[0] == 0) {
 next FORLOOP;
  }

$ctmp = log($carea[0])/log(10);
$ctmp =~ /^(\d*)/;
$ctmp = $1;
$carea[0] =~ /^(\d)(\d\d)/;
$carea[0] = "$1.$2e+$ctmp";

$tarea[0] += $carea[0];

for ($j = 1; $j<= @tech; $j++) {
$carea[$j] = $carea[0] * ($tech[$j-1]/2) * ($tech[$j-1]/2) /
1000000;

$carea[$j] =~ /(\d*)\.(\d\d\d)/;
$carea[$j] = "$1.$2";

$tarea[$j] += $carea[$j];
}
$~ = DataLine;
write;
}

$ctmp = log($tarea[0])/log(10);
$ctmp =~ /^(\d*)/;
$ctmp = $1;
$tarea[0] =~ /^(\d)(\d\d)/;
$tarea[0] = "$1.$2e+$ctmp";

for ($j = 1; $j<= @tech; $j++) {
```

```
$numcores[$j] = $chipFrac * $chipSize / $tarea[$j];
   $numcores[$j] =~ /(\d+)/;
   $numcores[$j] = $1;

}

$~ = TotalLine;
write;
$~ = NumLine;
write;

print "\n\n\n";



format TopLine =

Chip Area = @>>> mm2
$chipSize
------------------------------------------------------------------------------
Component Size    Num  L-Area   0.250u  0.180u  0.130u  0.070u  0.050u  0.035u
------------------------------------------------------------------------------
.
format DataLine =
@<<<<<<<<  @>> @>>>>> @>>>>>>> @>>>>>> @>>>>>>  @>>>>>  @>>>>>  @>>>>>  @>>>>>
$compname, $compsize, $compnum, $carea[0], $carea[1], $carea[2], $carea[3],
$carea[4], $carea[5], $carea[6]
.

format NumLine =

Number of Cores (@>>> Chip)    @>>>>>> @>>>>>>  @>>>>>  @>>>>>  @>>>>>  @>>>>>
$chipFrac,$numcores[1], $numcores[2], $numcores[3], $numcores[4], $numcores[5]
, $numcores[6]
------------------------------------------------------------------------------
.

format TotalLine =

Total Area            @>>>>>>> @>>>>>> @>>>>>>  @>>>>>  @>>>>>  @>>>>>  @>>>>>
$tarea[0], $tarea[1], $tarea[2], $tarea[3], $tarea[4], $tarea[5], $tarea[6]
.
```

References

[1] B. Baterman, C. Freeman, J. Halbert, K. Hose, G. Petrie, and E. Reese. A 450MHz 512kB second-level cache with a 3.6GB/s data bandwidth. *ISSCC*, pages 358–359, 1998. Intel Corporation, Hillsboro, OR.

[2] D. Carlson, A. Jain, P. Bannon, T. Benninghoff, M. Bertone, R. Blake-Campos, G. Bouchard, D. Brasili, R. Castelino, B, Lily, S. Mehta, B. Miler, R. Mueller, M. Nagarajan, A. Olesin, V. Yalala, Y. Saito, A. Chen, H. Kobayashi, S. Kobayashi, SB. Park, GC. Hwang, KI. Kim, and SJ. Kim. A 667MHz RISC microprocessor containing a 6.0ns 64b integer multiplier. *ISSCC*, pages 294–295, 1998. Digital Semiconductor, Digital Equipment Corp., Hudson, MA, Mitsubishi Eletric Corp., Japan, Samsung Electronics Corp., Japan.

[3] Keith Diefendorff. K7 challenges Intel. New AMD processor could beat Intel's Katmai. *Microprocessor Report*, page 1, October 26 1998.

[4] Keith Diefendorff. PentiumIII = PentiumII + SSE. Internet SSE architecture boosts multimedia performance. *Microprocessor Report*, page 1, March 8 1999.

[5] Keith Diefendorff. Sony's emotially charged chip. Killer floating point "Emotion Engine" to power playstation 2000. *Microprocessor Report*, page 1, April 19 1999.

[6] Peter N. Glaskowsky. MAP1000 unfolds at Equator. New media processor covers a lot of territory. *Microprocessor Report*, page 1, December 7 1998.

[7] Linley Gwennap. G4 is first PowerPC with AltiVec. Due mid-1999, Motorola's next chip aims at Macintosh, networking. *Microprocessor Report*, page 17, November 16 1998.

[8] Y. Hagihara, S. Inui, A. Yoshikawa, S. Nakazato, S. Iriki, R. Ikeda, Y. Shibue, T. Inaba, M. KAgamihara, and M. Yamashina. A 2.7ns 0.25um CMOS 54x54b multiplier. *ISSCC*, pages 296–297, 1998. NEC Corporation, Japan.

[9] Tom R. Halfhill. GP1000 has rewritable microcode. Imsys processor executes Java bytecodes and concurrent microcode processes. *Microprocessor Report*, page 14, December 28 1998.

[10] Atsuki Inoue, Ryoichi Ohe, Shoichiro Kashiwakura, Shin Mitarai, Takayuki Tsuru, Tetsuo Izawa, and Gensuke Goto. A 4.1ns compact 54x54b multiplier utilizing sign select booth encoders. *ISSCC*, pages 416–417, 1997. Fujitsu, Japan.

[11] H. Kubosawa, H. Takahashi, S. Ando, Y. Asada, A. Asato, A. Suga. M. Kimura, N. Higaki, H. Miyake, T. Sato, H. Anbutsu, T. Tsuda, T. Yoshimura, I. Amano, M. Kai, and S. Mitarai. A 1.2W 2.16GOPS/720MFLOPS embedded superscalar microprocessorfor multimedia applications. *ISSCC*, pages 290–291, 1998. Fujitsu Laboratories Ltd., Japan.

[12] O. Nishii, F. Arakawa, K. Isibashi, S. Nakano, T. Shimura, K. Suzuki, M. Tachibana, Y. Totsuka, T. Tsunoda, K. Uchiyama, T. Yamada, T. Hattori, H. Maejima, N. Nakagawa, S. Narita, M. Seki, Y. Shimazaki, R. Satomura, T. Takasuga, and A. Hasegawa. A 200MHz 1.2W 1.4GFLOPS microprocessor with graphic operation unit. *ISSCC*, pages 288–289, 1998. Hitachi Ltd., Japan.

[13] N. Rohrer, C. Akrout, M. Canada, D. Cawthron, B. Davari, R. Floyd, S. Geissler, R. Goldblatt, R. Houle, P. KArtschoke, D. Kramer, P. McCormick, G. Salem, R. Schulz, L. Su, and L. Whitney. A 480MHz RISC microprocessor in 0.12um L_{eff} CMOS technology with copper interconnects. *ISSCC*, pages 240–241, 1998. IBM Microelectronics.

[14] S. Santhanam, A. Baum, D. Bertucci, M. Braganza, K. Broch, T. Broch, J. Burnette, E. Chang, K. Chui, D. Dobberpuhl, P. Donahue, J. Grodstein, I. Kim, D. Murray, M. Pearce, A. Silveria, D. Soudalay, A. Spink, R. Stepanian, A. Varadharajan, and R. Wen. A low-cost 300MHz RISC CPU with attached media processor. *ISSCC*, pages 298–299, 1998. Digital Equipment Corporation, Palo Alto, CA.

[15] J. Silberman, N. Aoki, D. Boerstler, J. Burns, S. Dhong, A. Essbaum, U. Goshal, D. Heidel, P. Hofstee, K. Lee, D. Meltzer, H. Ngo, K. Nowka, S. Posluszny, O. Takahashi, I. Vo, and B. Zoric. A 1.0GHz single-issue 64b PowerPC integer processor. *ISSCC*, pages 230–231, 1998. IBM Austin Research Laboratory, Austin, TX.

[16] S. Storino, A Aipperspach, J. Borkenhagen, R. Eichemeyer, S. Kunkel, S. Levenstein, and G. Uhlmann. A commmercial multi-threaded RISC processor. *ISSCC*, pages 234–235, 1998. IBM Corp., Rochester, MN.

[17] Jim Turley. IDT retools midrange MIPS devices. RC64474,'475 revamp midrange MIPS lineup; compete with NEC, QED. *Microprocessor Report*, page 10, October 5 1998.

[18] Jim Turley. MIPS RM7000 emerger from QED. Two levels of on-chip cache, L3 cache controller, suit chip for high end. *Microprocessor Report*, page 12, August 3 1998.

[19] T. Yabe, S. Miyano, K. Sato, M. Wada, R. Haga, O. Wada, M. Enkaku, T. Hojyo, K. Mimoto, M. Tazawa, T. Ohkubo, and K. Numata. Toshiba Corporation. A configurable DRAM macro design for 2112 derivative organisations to be synthesized using a memory generator. *ISSCC*, pages 72–73, 1998.