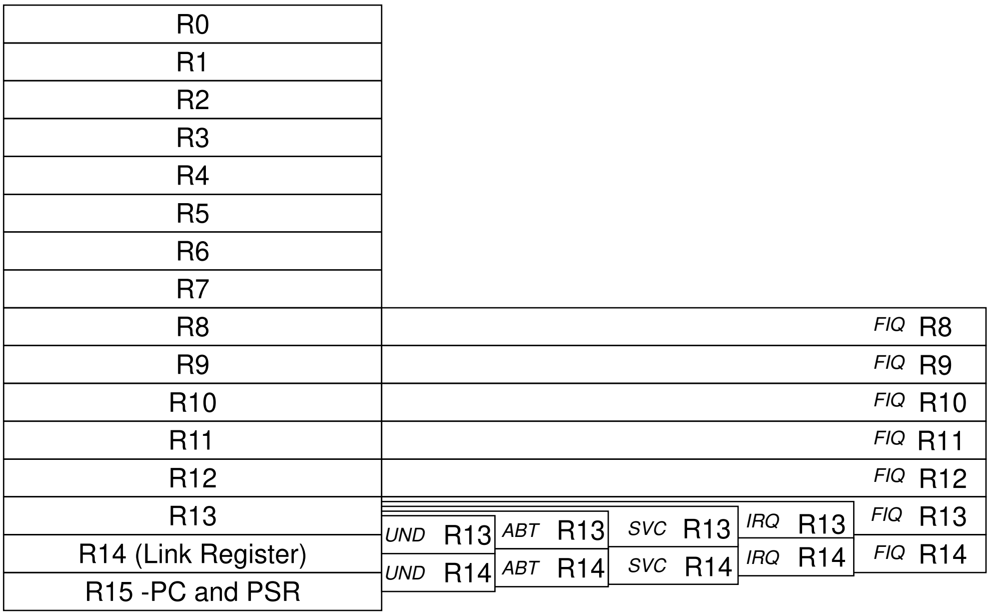

Figure B-4: The ARM6 register organization

The general functionality of the processor remains the same as the ARM2. There is a special mode flag that allows the ARM6 to operate as a 26-bit ARM2. This flag can be set either by strapping a hardware pin or by changing the flag which appears in the CPSR by software. When running in 26-bit mode, the ARM6 has only four modes and 27 registers and a combined PC and PSR (as described in the ARM2 section). The ARM6 is described in more detail elsewhere [ARM91].

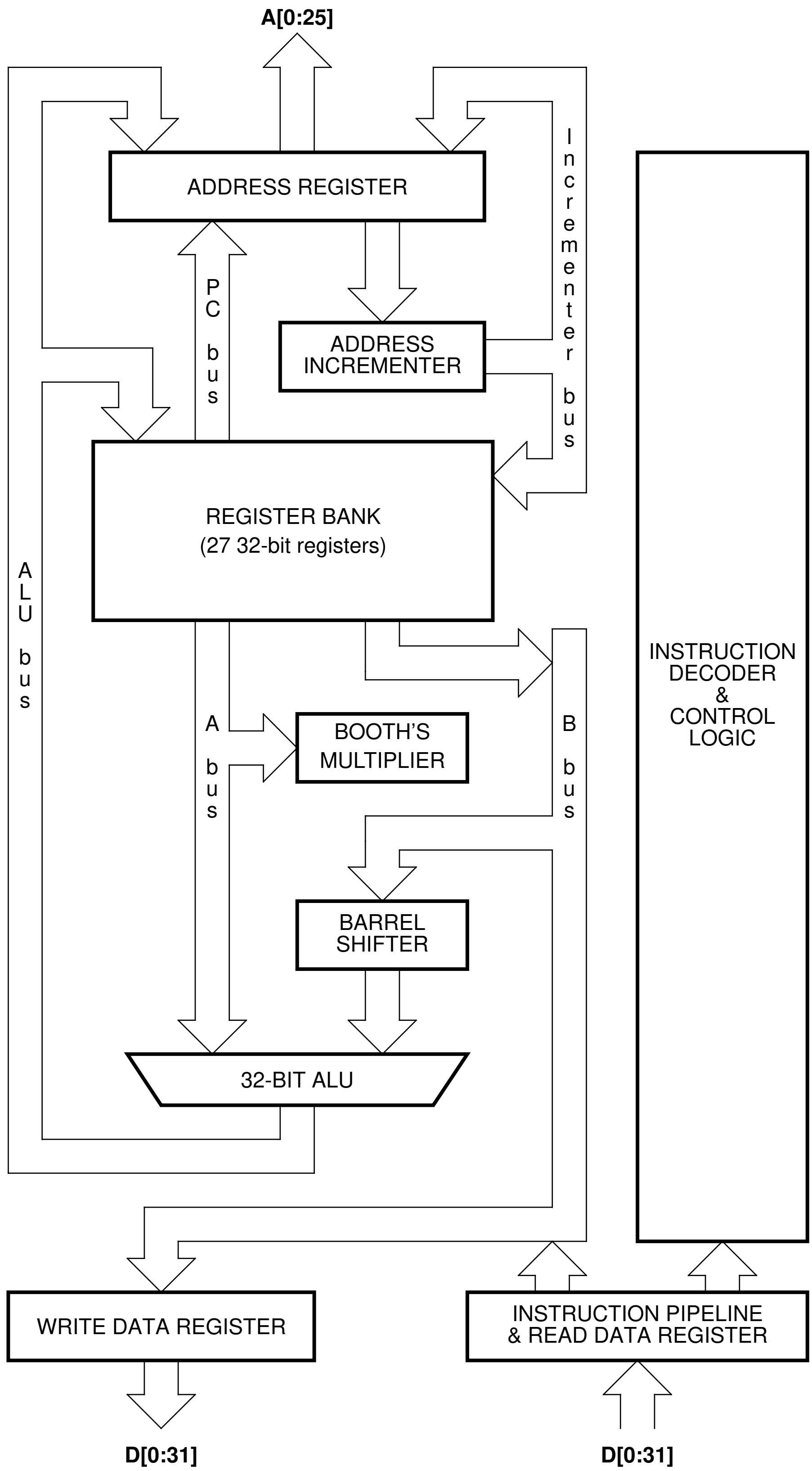

Figure B-3: ARM2 block diagram

configured such that various stack paradigms can be supported (e.g. empty/full ascending/descending stacks). These instructions are intended to make procedure entry/exit more efficient (i.e. saving and restoring registers).

The next class of instruction are branches. A special link flag is available to the programmer to cause the address of the following instruction to be placed in the link register (*R14*). This allows subroutine calls to be made. To return from a subroutine entered in this manner, the contents of *R14* are used as the new program counter (i.e. move *R14* -> *R15*).

The remaining instruction classes cover supervisor calls and external coprocessors.

An unusual feature of the ARM2 is that all instructions are conditionally executed, so that short forward branches are usually unnecessary. The instruction set is described in more detail elsewhere [Cock87].

B.1.2 Organization

The internal architecture is 32-bit and is organized as shown in figure B-3. This shows the register bank described previously, the barrel shifter, a 2-bit Booth's multiplier and the ALU. The address incrementer is used to generate the sequential addresses during instruction prefetch and multiple register transfer instructions.The external data bus is also 32-bits but the address bus is only 26-bits thus giving an address range of 64 Mbyte.

The operation of the ARM2 is pipelined into three stages as follows:-

1. Instruction PreFetch
2. Instruction Decode
3. Execute

The execute stage is the complete datapath operation i.e. register read, shift, ALU operation and result writeback. Each stage takes one cycle to complete so that a new instruction may start every cycle.

The ARM2 is described in detail elsewhere [Furb89, VLSI90].

B.2 The ARM6

The ARM6 was developed to extend the 26-bit address range to 32 bits and to provide additional modes to ease operating system design. The two new modes each have a corresponding set of *R13/R14* registers so that the total number of registers has increased from 27 to 31, but still with only sixteen visible at any one time. The register structure is illustrated in figure B-4.

A consequence of the 32-bit program counter is that the processor status information can no longer be stored in the same register. The concept of a separate PSR has been introduced to solve this problem. There is the Current Processor Status Register (CPSR) which contains the working flags and there is a set of Saved Processor Status Registers (SPSRs) one for each of the privileged modes. This allows the status information to be saved across mode changes. Two new instructions have been added to allow the status information to be transferred to and from the CPSR and SPSR (of the current mode) to one of the general purpose registers. The format of the new CPSR/SPSR is similar to the format of the ARM2 status flags, with the addition of 3 extra mode bits and with the I/F flags moved from bits 27/26 (as shown in figure B-2) to bits 7/6.

PC and PSR can be treated as a single entity for saving and loading but the PC can be used separately when it is used as a base address.

B.1.1 Instruction set

The ARM2 instruction set is based on a load/store model. There are six classes of instructions. The first is data operations; these perform arithmetic and logical operations on the register contents. The status flags of the processor can be changed according to the result of the operation if the 'set flags' bit in the instruction is true. This allows the programmer to determine which instructions can change the status flags. The ARM2 also allows all the data operations (except multiplies) to shift, by an arbitrary amount, one of the operands before performing the operation so there is no separate shift operation.

The second class of instructions loads and stores data which transfer data between memory and the register bank. The address in memory is calculated from the contents of a register (the base) and the addition of either a second register or a 12-bit immediate value (the offset). The contents of the register containing the offset may optionally be shifted before the address is calculated. The instruction class also supports auto increment/decrement of the base register with the option of specifying whether the change should be before or after the base is used to calculate the current memory address (i.e. pre-/post-increment/ decrement).

The third class of instruction allows multiple registers to be transferred to or from memory by a single instruction. The registers to be transferred are specified by the programmer. The registers are stored to memory in a sequential manner at an address determined by the contents of a base register. The base register is modified after the operation. How the base is modified and where exactly the registers are stored in relation to the base address can be

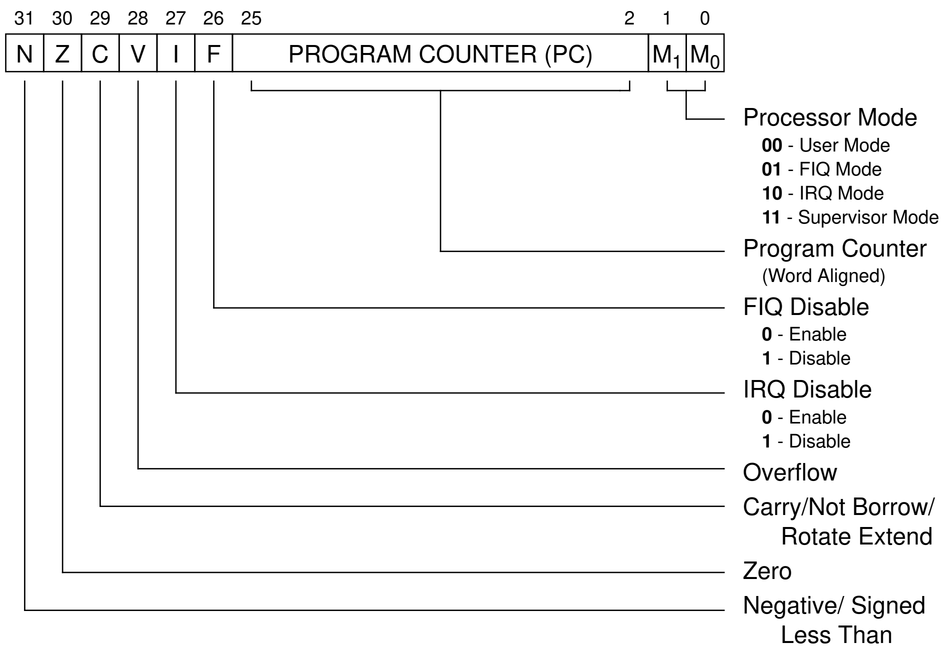

Figure B-2: Program counter and program status word

Appendix B: The ARM processor

The Advanced RISC Machine (**ARM**) is a 32-bit general purpose reduced instruction set microprocessor. The 1987 ARM2 will be described first, followed by the modifications that have lead to the ARM6.

B.1 The ARM2

The ARM2 features 27 registers each of 32 bits. The registers are organized into a set of partially overlapping banks of registers. There are fifteen general purpose registers and a register containing both the program counter and the program status register (PSR) available at any time. The ARM2 supports four modes - User, Supervisor (*SVC*), Interrupt (*IRQ*) and Fast Interrupt (*FIQ*). Figure B-1 shows how the registers banks are mapped on to the four modes.

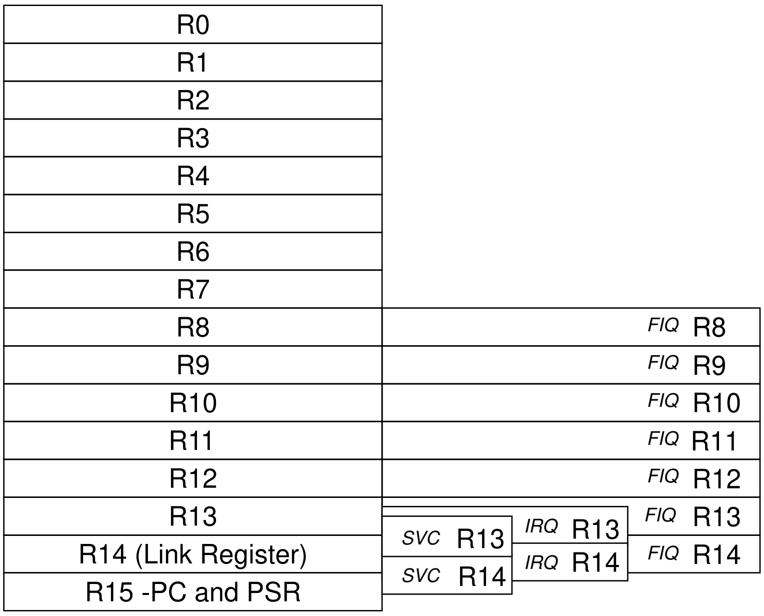

Figure B-1: The ARM2 register organization

The external address bus of the ARM2 is only 26-bits wide. This gives an addressing range of 64 MBytes. A consequence of this is that the program counter needs only to cover this range. Instructions in the ARM2 are always word aligned so in fact only 24-bits are needed for the program counter. The remaining bits in the 32-bit word are used to store the processor status bits. The format of the combined PC and PSR is shown in figure B-2. The

SPICE timings for: TLTCHR

Change history

version	author	date	comment
0.1	P.Day	5/5/1993	initial version

Signals

There are 3 input signals *In, En, NEn,* and the reset signal *Cdn*. There are two outputs *Out* and *nOut.* It should be noted that the complimentary output has poor drive capabilities.

In -> Out (nOut)

Load	Stand. Load	I/P Fall Prop.	I/P Rise Prop.	Rise	Fall
0.06	1	2.51 (1.84)	2.08 (1.08)	1.61 (2.27)	1.25 (2.98)
0.12	2	2.72 (1.85)	2.32 (1.09)	2.05 (2.15)	1.45 (3.02)
0.18	4	2.90 (1.84)	2.54 (1.09)	2.53 (2.12)	1.69 (3.06)
0.24	5	3.07 (1.84)	2.76 (1.09)	2.97 (2.10)	1.97 (3.06)
0.50	10	3.73	3.69	4.89	3.07
1.00	20	4.79	5.45	8.75	5.17

En/nEn -> Out (nOut)

Load	Stand. Load	En -> O/P low	En -> O/P high	Rise	Fall
0.06	1	2.21 (1.58)	1.79 (0.82)	1.67 (2.11)	1.44 (2.87)
0.12	2	2.41 (1.57)	2.01 (0.82)	2.16 (1.96)	1.72 (2.92)
0.18	4	2.62 (1.58)	2.25 (0.83)	2.54 (1.94)	1.88 (2.95)
0.24	5	2.80	2.49	2.95	2.10
0.50	10	3.48	3.42	4.94	3.06
1.00	20	4.65	5.19	8.78	5.20

Capacitance

In	0.029pF		
En	0.035pF	nEn	0.058pF
Out	0.132pF	nOut	0.143pF
Cdn	0.054pF		

SPICE timings for: SELECT2

Change history

version	author	date	comment
0.1	N.C.Paver	5/5/1993	initial version

Signals

There are 2input signals *In1* and *Select* and the reset signal *Cdn*. There are two outputs *True* and *False and* their complements *nTrue* and *nFalse.* The measurements are for equal loads on both true and complement outputs.

True/False

Load	Stand. Load	Fall Prop.	Rise Prop	Rise	Fall
0.06	1	3.85	4.73	1.86	1.46
0.12	2	4.63	5.71	2.44	1.85
0.18	4	5.40	6.70	3.09	2.22
0.24	5	6.16	7.63	3.47	2.62
0.50	10	9.52	11.83	6.06	4.24
1.00	20	15.64	19.71	10.30	7.48

nTrue/nFalse

Load	Stand. Load	Fall Prop.	Rise Prop	Rise	Fall
0.06	1	3.56	2.95	2.77	2.36
0.12	2	4.18	3.50	3.26	2.86
0.18	4	4.83	4.05	3.82	3.43
0.24	5	5.48	4.58	4.39	3.97
0.50	10	8.19	6.93	6.75	6.28
1.00	20	13.38	11.17	11.31	10.30

Capacitance

In	**0.168PF**
Sel	**0.181PF**
True/False	**0.194PF**
nTrue/NFalse	**0.278PF**
Cdn	**0.227PF**

Blank

Load	Stand. Load	Fall Prop.	Rise Prop	Rise	Fall
0.12	2	2.43	3.44	1.54	1.22
0.18	4	2.58	3.69	1.92	1.43
0.24	5	2.72	3.80	2.19	1.63
0.50	10	3.28	4.51	3.53	2.54
1.00	20	4.28	5.92	6.32	4.42

nBlank

Load	Stand. Load	Fall Prop.	Rise Prop	Rise	Fall
0.06	1	2.41	1.69	2.88	2.25
0.12	2	2.41	1.69	2.90	2.28
0.18	4	2.43	1.66	2.88	2.28
0.24	5	2.41	1.64	2.85	2.15
0.50	10	2.39	1.69	2.86	2.06
1.00	20	2.46	1.69	2.86	2.23

Capacitance

In **0.213PF**

Dot **0.182PF**

nDot **0.176PF**

Blank **0.109PF**

nBlank **0.232PF**

Cdn **0.186PF**

Blank

Load	Stand. Load	Fall Prop.	Rise Prop	Rise	Fall
1.00	20	10.32	11.97	8.62	7.04

nBlank

Load	Stand. Load	Fall Prop.	Rise Prop	Rise	Fall
0.06	1	2.62	1.96	3.44	2.78
0.12	2	2.93	2.22	3.97	3.25
0.18	4	3.12	2.58	4.62	3.77
0.24	5	3.36	2.86	5.33	4.19
0.50	10	4.41	4.02	7.90	6.38
1.00	20	6.45	6.46	13.12	10.95

Measurements taken with only *Dot* and *Blank* loaded (Temp=0C):

Dot

Load	Stand. Load	Fall Prop.	Rise Prop	Rise	Fall
0.06	1	2.61	2.19	1.60	1.10
0.12	2	2.71	2.35	1.93	1.32
0.18	4	2.89	2.52	2.22	1.54
0.24	5	3.06	2.70	2.54	1.66
0.50	10	3.58	3.40	3.94	2.64
1.00	20	4.50	4.73	6.69	4.42

nDot

Load	Stand. Load	Fall Prop.	Rise Prop	Rise	Fall
0.06	1	1.17	1.89	2.35	2.30
0.12	2	1.17	1.84	2.27	2.20
0.18	4	1.17	1.86	2.02	2.17
0.24	5	1.17	1.91	2.00	2.17
0.50	10	1.17	1.91	1.95	2.17
1.00	20	1.17	1.84	1.92	2.17

Blank

Load	Stand. Load	Fall Prop.	Rise Prop	Rise	Fall
0.06	1	2.24	3.22	1.19	1.19

SPICE timings for: TOGGLE

Change history

version	author	date	comment
0.1	N.C.Paver	5/5/1993	initial version

Signals

There is a single input *In* and the reset signal *Cdn*. There are two outputs *Dot* and *Blank and* their complements *nDot* and *nBlank.* The measurements are for equal loads on both true and complement outputs.

Dot

Load	Stand. Load	Fall Prop.	Rise Prop	Rise	Fall
0.06	1	2.97	2.52	1.62	1.26
0.12	2	3.42	3.07	2.00	1.59
0.18	4	3.98	3.61	2.47	1.97
0.24	5	4.39	4.16	3.00	2.28
0.50	10	6.63	6.49	5.15	3.80
1.00	20	10.74	10.89	9.06	7.01

nDot

Load	Stand. Load	Fall Prop.	Rise Prop	Rise	Fall
0.06	1	1.46	2.19	2.87	2.80
0.12	2	1.70	2.44	3.42	3.21
0.18	4	1.95	2.75	4.15	3.64
0.24	5	2.19	3.00	4.62	4.06
0.50	10	3.21	4.31	7.39	6.27
1.00	20	5.19	6.64	12.61	10.6

Blank

Load	Stand. Load	Fall Prop.	Rise Prop	Rise	Fall
0.06	1	2.57	3.58	1.43	1.06
0.12	2	3.06	4.13	1.80	1.50
0.18	4	3.57	4.66	2.40	1.86
0.24	5	4.10	5.16	2.70	2.27
0.50	10	6.17	7.47	4.66	3.83

SPICE timings for: MULLC3R

Change history

version	**author**	**date**	**comment**
0.1	N.C.Paver	5/5/1993	initial version

Signals

There are 3 input signals *In1, In2, In3,* and the reset signal *Cdn*. There are two outputs *Out* and *nOut.* Only the true output should be used as the complimentary output has poor drive capabilities.

Load	Stand. Load	Fall Prop.	Rise Prop	Rise	Fall
0.06	1	2.88	2.27	1.64	1.34
0.12	2	3.11	2.53	2.11	1.64
0.18	4	3.33	2.77	2.59	1.96
0.24	5	3.53	3.01	3.02	2.29
0.50	10	4.28	3.98	5.03	3.42
1.00	20	5.54	5.76	8.71	5.60

Capacitance

In1/In2/In3 **0.05PF**

Out **0.129PF**

Cdn **0.054PF**

SPICE timings for: MULLC2

Change history

version	author	date	comment
0.1	N.C.Paver	5/5/1993	initial version

Signals

There are 2 input signals *In1,* and *In2*. There are two outputs *Out* and *nOut.* Only the true output should be used as the complimentary output has poor drive capabilities.

Load	Stand. Load	Fall Prop.	Rise Prop	Rise	Fall
0.06	1	2.13	1.62	1.48	1.36
0.12	2	2.33	1.86	1.92	1.60
0.18	4	2.53	2.09	2.26	1.78
0.24	5	2.68	2.30	2.77	2.01
0.50	10	3.35	3.24	4.75	2.97
1.00	20	4.52	4.99	8.57	5.12

Capacitance

In1/In2 **0.05PF**

Out **0.129PF**

Cdn **0.054PF**

SPICE timings for: MULLC2R

Change history

version	author	date	comment
0.1	N.C.Paver	5/5/1993	initial version

Signals

There are 2 input signals *In1, In2,* and the reset signal *Cdn*. There are two outputs *Out* and *nOut.* Only the true output should be used as the complimentary output has poor drive capabilities.

Load	Stand. Load	Fall Prop.	Rise Prop	Rise	Fall
0.06	1	2.31	2.02	1.61	1.42
0.12	2	2.51	2.24	2.06	1.68
0.18	4	2.72	2.46	2.56	1.78
0.24	5	2.87	2.69	2.92	1.92
0.50	10	3.53	3.61	4.79	2.99
1.00	20	4.72	5.37	8.59	5.20

Capacitance

In1/In2	**0.05PF**
Out	**0.129PF**
Cdn	**0.054PF**

SPICE timings for: DWAIT2

Change history

version	author	date	comment
0.1	N.C.Paver	5/5/1993	initial version

Signals

There are three input signals *A1*, *A2*, *Fire* and the reset *Cdn*. There are two outputs *Z1* and *Z2* and their complements *nZ1* and *nZ2*. The measurements are taken with equal loads on true and complement outputs.

Z1/Z2

Load	Stand. Load	Fall Prop.	Rise Prop	Rise	Fall
0.06	1	3.51	4.22	1.81	1.39
0.12	2	3.97	4.73	2.32	1.76
0.18	4	4.40	5.21	2.85	2.15
0.24	5	4.83	5.68	3.37	2.53
0.50	10	6.70	7.69	5.71	4.23
1.00	20	10.08	11.54	10.17	7.42

nZ1/nZ2

Load	Stand. Load	Fall Prop.	Rise Prop	Rise	Fall
0.06	1	3.11	2.64	2.59	2.18
0.12	2	3.28	2.89	3.03	2.51
0.18	4	3.45	3.07	3.49	2.81
0.24	5	3.60	3.28	3.91	3.11
0.50	10	4.29	4.19	5.92	4.72
1.00	20	5.54	5.71	9.78	7.56

Capacitance

A1/A2	**0.098PF**
Z1/Z2	**0.183PF**
nZ1/nZ2	**0.274PF**
Cdn	**0.226PF**

SPICE timings for: DXor/DXNor

Change history

version	author	date	comment
0.1	N.C.Paver	5/5/1993	initial version

Signals

There are 4 input signals *In1, In2, nIn1, nIn2*. There is a single output *Out.*

Load	Stand. Load	Fall Prop.	Rise Prop	Rise	Fall
0.06	1	0.63	1.31	2.01	1.48
0.12	2	0.98	1.68	2.74	1.96
0.18	4	1.31	2.05	3.52	2.47
0.24	5	1.57	2.42	4.29	2.86
0.50	10	2.50	4.10	7.95	4.65
1.00	20	4.25	7.28	15.18	8.51

Capacitance

In1/In2 0.038PF

SPICE timings for: DMULLC2

Change history

version	author	date	comment
0.1	N.C.Paver	5/5/1993	initial version

Signals

There are 2 input signals *In1, In2,* and the reset signal *Cdn*. There are two outputs *Out* and *nOut.* The measurements are for equal loads on both outputs.

Out

Load	Stand. Load	Fall Prop.	Rise Prop	Rise	Fall
0.06	1	2.39	2.09	1.71	1.28
0.12	2	2.86	2.57	2.11	1.54
0.18	4	3.28	3.06	2.61	1.94
0.24	5	3.73	3.55	3.13	2.31
0.50	10	5.55	5.61	5.46	3.88
1.00	20	9.10	9.50	9.73	6.98

nOut

Load	Stand. Load	Fall Prop.	Rise Prop	Rise	Fall
0.06	1	1.14	1.73	2.78	2.01
0.12	2	1.32	1.96	3.68	2.32
0.18	4	1.49	2.14	3.60	2.69
0.24	5	1.66	2.35	3.95	2.99
0.50	10	2.32	3.18	5.85	4.35
1.00	20	3.61	4.83	9.44	7.23

Capacitance

In1/In2 **0.09PF**

Out **0.134PF**

nOut **0.226PF**

Cdn **0.105PF**

R1/R2 -> R

Load	Stand. Load	Fall Prop.	Rise Prop	Rise	Fall
0.50	10	2.50	4.10	7.95	4.65
1.00	20	4.25	7.28	15.18	8.51

Capacitance

R	**0.071PF**
R1	**0.145PF**
R2	**0.186PF**
nR1/nR2	**0.05PF**
D	**0.170PF**
D1/D2	**0.184PF**
nD1/nD2	**0.275PF**
Cdn	**0.227PF**

SPICE timings for: CALL2

Change history

version	author	date	comment
0.1	N.C.Paver	5/5/1993	initial version

Signals

There are three input signals *R1*, *R2*, *D* and the complements *nR1* and *nR2*. There is also a reset *Cdn*. There are three outputs *D1*, *D2*, *R* and their complements *nD1* and *nD2.* The forward measurement (*R1/R2 -> R)* is equivalent to a DXOR and the reverse measurement (*D -> D1/D2)* is the same as a DWAIT2. The measurements shown below are taken from the appropriate corresponding gate.

D -> D1/ D2 (equal load on *D1/nD1* and *D2/nD2)*

Load	Stand. Load	Fall Prop.	Rise Prop	Rise	Fall
0.06	1	3.51	4.22	1.81	1.39
0.12	2	3.97	4.73	2.32	1.76
0.18	4	4.40	5.21	2.85	2.15
0.24	5	4.83	5.68	3.37	2.53
0.50	10	6.70	7.69	5.71	4.23
1.00	20	10.08	11.54	10.17	7.42

D- > nD1/nD2 (equal load on *D1/nD1* and *D2/nD2*)

Load	Stand. Load	Fall Prop.	Rise Prop	Rise	Fall
0.06	1	3.11	2.64	2.59	2.18
0.12	2	3.28	2.89	3.03	2.51
0.18	4	3.45	3.07	3.49	2.81
0.24	5	3.60	3.28	3.91	3.11
0.50	10	4.29	4.19	5.92	4.72
1.00	20	5.54	5.71	9.78	7.56

R1/R2 -> R

Load	Stand. Load	Fall Prop.	Rise Prop	Rise	Fall
0.06	1	0.63	1.31	2.01	1.48
0.12	2	0.98	1.68	2.74	1.96
0.18	4	1.31	2.05	3.52	2.47
0.24	5	1.57	2.42	4.29	2.86

Appendix A: Timing characteristics

This appendix contains the timing characteristics of some of the Micropipeline library elements designed for use in the asynchronous ARM. Each cell is characterised by simulating the extracted netlist with SPICE under worst case conditions for a selection of loads. The exact conditions used are described in the next section. The cells analysed are listed below:

CALL2	- 2 input call block
DMULLC2	- 2 input C-Gate with double strength drive on the input stack to enable the gate to drive from the internal node
DXOR/DXNOR	- 2 input XOR, each input requires complimentary
DWAIT2	- 2 input Decision-Wait
MULLC2R	- 2 input C-Gate with reset
MULLC2	- 2 input C-Gate without reset
MULLC3R	- 2 input C-Gate with reset
TOGGLE	- TOGGLE element
SELECT2	- 2 input SELECT block
TLTCHR	- transparent latch with true & complement enable required

A.1 Measurement conditions

The measurements are taken at the following conditions:

VDD	**4.6**	**VSS**	**0.1**
Temp	**100^{o}C**		
Process	**CMOS 1.2 micron, Slow-Slow transistors**		

All measurements in the tables are in nanoseconds (unless otherwise stated)

Inputs ramp 0-100% (0-4.6v) in 4 nS

The propagation delay is measured from the 50% point of the input to the 50% point on the output.

The rise/fall time is measured between the 10% and 90% value of the output.

[VLSI91] CMOS Lamda Layout Design Rules for Advanced Lithography. VLSI Technology Inc., San Jose, U.S.A., Document No: 02-ECLA-2, Confidential, December, 1991.

[Will90] Williams T.E., Latency and Throughput Tradeoffs in Self-Timed Asynchronous Pipelines and Rings. Technical Report CSL-TR-90-431, Computer Systems Laboratory, Stanford University, August, 1990.

[Will91] Williams T.E., Horowitz M.A., A 160ns 54bit {CMOS} Division Implementation using Self-Timing and Symmetrically Overlapped {SRT} Stages. Proceedings of the 10th {IEEE} Symposium on Computer Arithmetic, pages 210-217. 1991

[Will91a] Williams T.E., Horowitz M.A., A Zero-Overhead Self-Timed 160-ns 54-b CMOS Divider. IEEE Journal of Solid-State Circuits. 26(11):1651-1661, November, 1991.

[XIDA87] Mainsail Language Manual. Volume 1, XIDAK Inc., Menlo Park, CA, U.S.A. 1987.

[Yako92] Yakovlev A.V., On Limitations and Extensions of STG Model for Designing Asynchronous Control Circuits. Proceedings of ICCD, pp. 396-400, 1992.

[Yant92] Yantchev J., An S-I Toggle Design. Private fax communication, 1992.

[Yun92] Yun K.Y., Dill D.L., Nowick S.M., Synthesis of 3D Asynchronous State Machines. 1992 IEEE International Conference on Computer Design: VLSI in Computers & Processors, pp. 346-350. October, 1992.

[Yun92b] Yun K.Y., Dill D.L., Nowick, S.M., Practical Generalizations of Asynchronous State Machines. Technical Report CSL-TR-92-544, Computer Systems Laboratory, Stanford University, July, 1992.

[Smit92] Smith L., A Guide to the ASIM Simulation and Modelling Package, ARM Ltd., 1992.

[Stal92] Stallman R.M., Using and Porting GNU CC. Free Software Foundation Inc., Cambridge, MA, U.S.A, 1992.

[Sun92] The SuperSPARC Microprocessor. Sun Microsystems Inc., Technical white paper, Moutain View, CA, U.S.A, 1992.

[Suth89] Sutherland I.E., Micropipelines. Communications of the ACM. 32(6):720-738, January, 1989.

[Suth86] Sutherland I.E., Sproull R.F. Asynchronous Systems. Sutherland, Sproull & Associates, Palo Alto, California, U.S.A, September, 1986.

[Sven89] Svensson C., Yuan J., High-Speed CMOS circuit techniques. IEEE Journal of Solid-State Circuits, 24(1):62-72, February, 1989.

[Theo93] Theodoropoulos G.K., Woods J.V., Modeling and Simulation of Asynchronous Computer Architectures. (to be published), Manchester University, 1993.

[TTL85] The TTL Data Book: Volume 1. Texas Instruments, 1985.

[Unge59] Unger S.H., Hazards and delays in asynchronous sequential switching circuits, IRE Trans. Circuit Theory, CT-6:12-25, March, 1959.

[Unge69] Unger S.H., Asynchronous Sequential Switching Circuits. Wiley-Interscience, John Wiley & Sons, Inc., New York, 1969.

[Vanb90b] Vanbekbergen P. et al, Optimized Synthesis of Asynchronous Control Circuits from Graph-theoretic Specifications. Proceedings of ICCAD, pp.184-187, 1990.

[Vanb90b] Vanbekbergen P. et al, Time & Area Perfomant Synthesis of Asynchronous Control Circuits. Proceedings of TAU'90, 1990.

[vBer88] van Berkel C.H., Rem M., Saeijs R.W.J.J., VLSI Programming. Proceedings of ICCD, IEEE, pp. 152-156, 1988.

[vBer88a] van Berkel C.H., Rem M., Saeijs R.W.J.J., Compilation of Communicating Processes into Delay-Insensitive Circuits. Proceedings of ICCD, IEEE, pp. 157-162, 1988.

[vBer91] van Berkel C.H., Kessels J., Roncken M., Saeijs R.W.J.J, Schalij F., The VLSI-programming langauge Tangram and its translation into handshake circuits. Proceedings of EDAC, pp. 384-389, 1991.

[vBer92] van Berkel C.H., Handshake circuits: an intermediary between communicating processes and VLSI, PhD thesis, Eindhoven Univeristy of Technology, 1992.

[vBer92a] van Berkel C.H., Beware the Isochronic Fork. INTEGRATION, the VLSI Journal. 13(2):103-128, June, 1992.

[VLSI90] Acorn Risc Machine (ARM) Family Data Manual. VLSI Technology Inc., Prentice Hall, Englewood Cliffs, New Jersey, 1990

[Myer93] Myers C., and Meng T.H-Y., Synthesis of Timed Asynchronous Circuits. 1993. (to appear in) IEEE Trans. on VLSI Systems, June '93.

[Nage73] Nagel L.W., Pederson D.O., Simulation Program with Integrated Circuit Emphasis (SPICE). Report ERL-M383, University of California, Berkeley, Electronics Research Lab., 1973.

[Nies88] Niessen C., van Berkel C.H., Rem M., Saeijs R.W.J.J., VLSI Programming and Silicon Compilation: A Novel Approach from Philips Research. Proceedings of ICCD, IEEE, pp. 150-151, 1988.

[Norw91] Nowick S.M., Dill D.L., Synthesis of Asynchronous State Machines Using a Local Clock. Proc. of 1991 Int. Conf. on Computer Design, pp. 192-197. IEEE Computer Society Press, October, 1991.

[Norw91b] Nowick S.M., Dill D.L., Automatic Synthesis of Locally-Clocked Asynchronous State Machines. Proc. of 1991 Int. Conf. on Computer-Aided Design, pp. 318-321. IEEE Computer Society Press, November, 1991.

[Norw92] Nowick S.M., Dill D.L., Asynchronous State Machine Synthesis Using a Local Clock. Proc. of 1991 MCNC Int. Workshop on Logic Synthesis. 1991.

[Pave91] Paver N.C., Condition Detection in Asynchronous Pipelines. UK Patent no 9114513, October 1991.

[Pave92a] Paver N.C., Day P., Furber S.B., Garside J.D., and Woods J.V., Register Locking in an Asynchronous Microprocessor. Proceedings of ICCD ‘92, pp 351-355, October 1992.

[Pave92b] Paver N.C., Micropipelines - Implementation. ACiD-WG/EXACT Workshop on Asynchronous Data Processing. Veldhoven, The Netherlands, December, 1992.

[Pave93] Paver N.C., Day P., A Micropipelined ARM. Unpublished presentation, IFIP working conference on Asynchronous Design Methodologies, Manchester, England, 1993

[Pete81] Peterson J., Petri net theory and modeling of systems. Prentice Hall, 1981.

[Rem90] Rem M., The Nature of Delay-Insensitive Computing. {IV} Higher Order Workshop, Banff 1990, pp. 105-122. Springer-Verlag, 1991.

[Saei88] Saeijs R.W.J.J., van Berkel C.H., The Design of the VLSI Image-Generator Zap. Proceedings of ICCD, IEEE, pp. 163-166, 1988.

[Seit70] Seitz C.L., Asynchronous Machines Exhibiting Concurrency. Record of the Project MAC Concurrent Parallel Computation. 1970.

[Seit80] Seitz C.L., System Timing. In Introduction to VLSI Systems, editors Mead C. A., Conway L. A., chapter 7, Addison-Wesley, 1980.

[Shoj88] Shoji M. CMOS Digital Circuit Technology. Prentice-Hall, Englewood Cliffs, New Jersey, 1988.

[Mart85b] Martin A.J., A Delay-insensitive Fair Arbiter. Technical Report 5193:TR:85, Computer Science Dept, Caltech, 1985.

[Mart86] Martin A.J., Compiling Communicating Processes into Delay-insensitive VLSI circuits. Distributed Computing, 1(4), 1986.

[Mart89] Martin A.J., Formal Program Transformations for VLSI Circuit Synthesis. UT Year of Programming Institute on Formal Developments of Programs and Proofs, editor E.W. Dijkstra, Addison-Wesley, Reading MA, 1989.

[Mart89a] Martin A.J., Burns S. M., Lee T. K., Borkovic D., Hazewindus P. J., Design of an Asynchronous Microprocessor. Advanced Research in VLSI 1989: Proceedings of the Decennial Caltech Conference on VLSI, ed. C. L. Seitz, MIT Press, pp 351-373, 1989.

[Mart89b] Martin A.J., Burns S. M., Lee T. K., Borkovic D., Hazewindus P. J., The first asynchronous microprocessor. Technical report CS-TR-89-06, Caltech, 1989.

[Mart90] Martin A.J. Synthesis of Asynchronous VLSI Circuits. Formal Methods for VLSI Design, editor J. Staunstrup, North-Holland, 1990.

[Mart90a] Martin A.J., The Limitations to Delay-Insensitivity in Asynchronous Circuits. Sixth MIT Conference on Advanced Research in VLSI, editor W. J. Dally, pp. 263-278. MIT Press, 1990.

[Mead80] Mead C., Conway L. Introduction to VLSI Systems. Addison-Wesley, London, 1980.

[Mehr92] Mehra R., Micropipelined Cache Design Strategies for an Asynchronous Microprocessor. MSc Thesis, Dept. of Computer Science, University of Manchester, 1992.

[Meng89] Meng T.H.Y., Brodersen R.W., Messerschmidtt D.G., Automatic Synthesis of Asynchronous Circuits from High-Level Specifications. IEEE Transactions on CAD, 8(11):1185-1205, November, 1989

[Mill65] Miller R.E. Sequential Circuits. Chapter 10, In Switching Theory, Wiley, New York, 1965.

[Moln83] Molnar C.E., Fang T-P., Synthesis of Reliable Speed-Independent Circuit Modules: I. General Method for Specification of Module-Environment Interaction and Derivation of a Circuit Realization. Technical Report 297, Computer Systems Laboratory, Institute for Biomedical Computing, Washington Univ., St. Louis, MO, 1983.

[Moln85] Molnar C.E., Fang T-P., Rosenberger F.U., Synthesis of Delay-Insensitive Modules. 1985 Chapel Hill Conference on Very Large Scale Integration, editor H. Fuchs, pp. 67-86. Computer Science Press, 1985.

[Myer92] Myers C., Meng T.H-Y., Synthesis of Timed Asynchronous Circuits. 1992 IEEE International Conference on Computer Design: VLSI in Computers & Processors, pp. 279-284. October, 1992.

[Joup93] Jouppi N.P. et al, A 300MHz 115W 32b Bipolar ECL Microprocessor. Symposium Record of Hot Chips V, Stanford University, CA, U.S.A., August, 1993.

[Kell74] Keller R.M., Towards a Theory of Universal Speed-Independent Modules. IEEE Transactions on Computers C-32(1):21-33, June 1974.

[Kern88] Kernighan B.W., Ritchie D.M., The C Programming Language. Prentice Hall, Englewood Cliffs, New Jersey, 1988.

[Kell93] Kelly R., Asynchronous Design Aspects of High Performance Logic. MSc Thesis in preparation, 1993.

[Kess90] Kessels J.L.W., Schalij F.D., VLSI Programming for the Compact Disc Player. Science of Computer Programming, 15:235-248, 1990.

[Kess90a] Kessels J.L.W., Rem M., Designing systolic, distributed buffers with bounded response time. Distributed Computing, 4:37-43, 1990.

[Kess91] Kessels J.L.W., The Systematic Design of a Systolic RSA Converter. Proceedings of the Workshop on Correct Hardware Design Methodologies, pp. 243-260, 1991.

[Kess92] Kessels J.L.W. et al, An Error Decoder for the Compact Disc Player as an Example of VLSI Programming. Proceedings of the European Conference on Design Automation, pp.69-74, 1992.

[Kinn76] Kinniment D.J., Woods J.V., Synchronisation and arbitration circuits in digital systems. Proc. IEE. 123(10):961-966, October, 1976.

[Lava92] Lavagno L., Sangiovanni-Vincentelli A., Linear Programming for Optimum Hazard Elimination in Asynchronous Circuits. 1992 IEEE International Conference on Computer Design: VLSI in Computers & Processors, pp. 275-278. October, 1992.

[Lin91] Lin K.J., Lin C.S., Automatic Synthesis of Asynchronous Circuits. Proceedings of DAC, pp. 296-301, 1991

[Lin91a] Lin K.J., Lin C.S., A Realization Algorithm of Asynchronous Circuits from STG. Proceedings of EDAC, pp.322-326, 1992

[Lin91a] Lin K.J., Lin C.S.,On the Verification of State-Coding in STGs, Proceedings of ICCAD, pp.118-122, 1992.

[Lind92] Linerholm, O., Aprika, S., Nadeau, M., The PC gets more personal. Byte, pp 128-136, July 1992.

[Mart85] Martin A.J., Distributed Mutual Exclusion on a Ring of Processes. Science of Computer Programming, 5:265-276, 1985.

[Mart85a] Martin A.J., The design of a Self-timed circuit for Distributed Mutual Exclusion. 1985 Chapel Hill Conference on VLSI, editor H. Fuchs, Computer Science Press, pp. 247-260, 1985.

[Gopa93a] Gopalakrishnan G., Akella, V., Specification, Simulation, and Synthesis of Self-Timed Circuits. 26th Hawaii Int. Conference on System Science (HICSS 1993), pp. 399-408. January, 1993.

[Hamb92] Hamburgen W.R., Fitch J.S, Packaging a 150W Bipolar Microprocessor. DEC Western Research Laboratory Research Report 92/1, 1992.

[Hauc93] Hauck S., Asynchronous Design Methodologies: An Overview. Dept. of Computer Science and Engineering, Technical Report 93-05-07, University of Washington, U.S.A. 1993.

[Hoar78] Hoare C.A.R, Communicating Sequential Processes. Communications of the ACM, 21(8):666-677, 1978.

[Hoar85] Hoare C.A.R., Communicating Sequential Processes. Prentice-Hall, 1985.

[Hors89] Horstmann J.U., Eichel H.W., Coates R.L. Metastability Behavior of CMOS ASIC Flip-Flops in Theory and Test. IEEE Journal of Solid-State Circuits. 24(1):146-157, February, 1989.

[Howa89] Howard D.W. *Dynamic PLAs*. ARM Ltd internal document, August, 1989.

[Huff64] Huffman D.A., The Synthesis of Sequential Switching Circuits. Sequential Machines: Selected Papers, editor Moore, E. F., Addison-Wesley, 1964.

[Inmo83] Occam Programming Manual. Inmos, Bristol, England, 1983.

[Jagg90] Jagger D.V., A Performance Study of the Acorn RISC Machine. M.Sc. Thesis, University of Canterbury, New Zealand, 1990.

[Jose90] Josephs M.B., Udding, J.T., Delay-Insensitive Circuits: An Algebraic Approach to their Design. Lecture Notes in Computer Science. editors J.C.M. Baeten and J.W. Klop, Volume 458, pp. 342-366. Springer-Verlag, 1990.

[Jose90a] Josephs M.B., Mak, R.H., Verhoeff, T., Asynchronous Design of a Router. Dept. of Math. and C.S., Eindhoven Univ. of Technology, September, 1990.

[Jose92] Josephs M.B. Speed-Independent Design of a Toggle. ACiD-WG/EXACT Workshop on Asynchronous Controllers and Interfacing. Leuven, Belgium, September, 1992.

[Jose91] Josephs M.B,. Udding, J.T. An Algebra for Delay-Insensitive Circuits. Workshop on Computer-Aided Verification, editors R.Kurshan and E. M. Clarke, pp. 147-175. AMS-ACM, 1991.

[Jose92a] Josephs M.B., et al, High-level Design of an Asynchronous Packet-routing Chip. In Designing Correct Circuits, editors J. Staunstrup and R. Sharp, IFIP transactions A:Computer Science and Technology, pp. 261-274, North Holland, 1992.

[Eshr89] Eshraghian K., Weste N. Principles of CMOS Design A Systems Perspective. Addison-Wesley, Wokingham, England, 1989.

[Farn93] Farnsworth C., A Brief Report on Handshake Circuits. ACID-WG Report, Manchester, 1993.

[Farn93a] Farnsworth C., AMULET group approach to design examples. ACID-WG workshop on Digital Signal Processing, Barcelona, 1993.

[Farn94] Farnsworth C., Low Power Implementations of an I^2C-Bus Expander, MSc Thesis, Dept. of Computer Science, Manchester University, 1994, to be submitted.

[Furb89] Furber S.B., VLSI RISC Architecture and Organization. Marcel Dekker, New York, 1989.

[Furb92] Furber S.B. Micropipelines - A Case Study. ACiD-WG/EXACT Workshop on Asynchronous Data Processing. Veldhoven, The Netherlands, December, 1992.

[Furb93a] Furber S.B, AMULET1 - An Asynchronous ARM Processor. Symposium Record of Hot Chips V, Stanford University, CA, U.S.A., August, 1993.

[Furb93b] Furber S.B., Day P., Garside J.D., Paver N.C., Woods J.V., A Micropipelined ARM. Proceedings of VLSI '93, Grenoble, France, September 1993, best paper award.

[Furb93c] Furber S.B., Computing without Clocks. Proceedings of the VII Banff Workshop: Asynchronous Hardware Design, Banff, Canada, 1993.

[Furb94] Furber S.B., Day P., Garside J.D., Paver N.C., Woods J.V., "AMULET1: A Micropipelined ARM", IEEE CompCon '94, San Francisco, March 1994.

[Gars92] Garside J.D. Micropipeline Structures. ACiD-WG/EXACT Workshop on Asynchronous Data Processing. Veldhoven, The Netherlands, December, 1992.

[Gars93] Garside J.D. A CMOS VLSI Implementaion of an Asynchronous ALU. Proceedings of the IFIP working conference on Asynchronous Design Methodologies, Manchester, England, 1993.

[Gopa90] Gopalakrishnan G., Jain P., Some Recent Asynchronous System Design Methodologies. Technical Report Number UU-CS-TR-90-016, Dept. of Computer Science, University of Utah, October, 1990.

[Gopa92] Gopalakrishnan G., Liebchen A., Dynamic Reordering of High Latency Transactions Using a Modified Micropipeline. 1992 IEEE International Conference on Computer Design: VLSI in Computers & Processors, pp. 336-340. October, 1992.

[Gopa93] Gopalakrishnan G., Some Unusual Micropipeline Circuits. Technical Report UU-CS-TR-93-015, Dept. of C.S., Univ. of Utah, July, 1993.

[Comp91d] QTV Timing Verifier. In Verification Tools, Compass Design Automation Inc., San Jose, U.S.A., Document No: VSD 41400. 1991.

[Cour75] Couranz G.R. and Wann D.F. Theoretical and Experimental Behavior of Synchronizers Operating in the Metastable Region. IEEE Transactions on Computers. C-24(6):604-616, June, 1975.

[Davi93] Davis A., Practical Asynchronous Design. Proceedings of the VII Banff Workshop: Asynchronous Hardware Design, Banff, Canada, 1993.

[Davi94] Davies R. Powertool Manual, AMULET group report, Manchester University, 1994

[Day92] Day P., A Micropipelined Multiplier. ACiD-WG/EXACT Workshop on Asynchronous Data Processing. Veldhoven, The Netherlands, December, 1992.

[Day93] Day P., Investigations into Micropipeline Latch Design Styles, (to be published), University of Manchester, 1993.

[DEC92] Alpha Architecture Handbook. Digital Equipment Corporation, Maynard, MA, U.S.A, 1992.

[DEC92a] DECChip 21064-AA RISC Microprocessor Preliminary Data Sheet. Digital Equipment Corporation, Maynard, MA, U.S.A, 1992.

[Dijk76] Dijkstra E.W., A Discipline of Programming. Prentice-Hall, Englewood Cliffs NJ, 1976.

[Dill89] Dill D.L., ACM Distinguished Dissertations: Trace Theory for Automatic Hierachical Verification of Speed-Independent Circuits. MIT Press, 1989.

[Dobb92] Dobberpuhl D. et al, A 200 MHz 64b Dual-Issue CMOS Microprocessor. IEEE Journal of Solid-State Circuits, 27(11):155-1565, November, 1992.

[Eber91] Ebergen J.C., A Formal Approach to Designing Delay-Insensitive Circuits. Distributed Computing. 5(3):107-119, 1991.

[Eber92] Ebergen J.C., Peeters A.M.G., Modulo-N Counters: Design and Analysis of Delay-Insensitive Circuits. 2nd Workshop on Designing Correct Circuits, Lyngby, editors J. Staunstrup, R. Sharp, pp. 27-46. Elsevier Science Publishers, June, 1992.

[Edwa92] Edwards D.A. Pitfalls in Asynchronous Design. AMULET Group Internal Report, 1992.

[Edwa93] Edwards D.A., Design and Implementation of I2C. ESPRIT 6143 - EXACT No: EXACT/MU/93Sept/C.1-Deliverable, Confidential, 1993.

[Ende93] Endecott P.B., Processor Architectures for Power Efficiency and Asynchronous Implementation. MSc Thesis, Dept. of Computer Science, University of Manchester, 1993.

[Burn87] Burns S.M., Automated Compilation of Concurrent Programs into Self-timed Circuits. MSc thesis, Computer Science Department, Caltech, 1987.

[Burn88] Burns S.M., Martin A.J., Synthesis of self-timed circuits by program transformation. In The Fusion of Hardware Design and Verification, pp. 99-116, editor G.J. Milne, Elsevier Science Publishers, 1988.

[Chan73] Chaney T.J., Molnar, C.E., Anomalous Behavior of Synchronizer and Arbiter Circuits. IEEE Transactions on Computers. C-22(4):421-422, April, 1973.

[Chu85], Chu T.A., Leung C.K.C., Wanuga T.S., A Design Methodology for Concurrent VLSI systems. Proceedings of ICCD, pp 407-410, 1985.

[Chu86a] Chu T.A., On the Models for Designing VLSI Asynchronous Digital Systems". INTEGRATION the VLSI Journal , Vol. 4, pp. 99-113, 1986.

[Chu86b] Chu T.A., Glasser L.A., Synthesis of Self-timed Control Circuits from Graphs : An Example. Proceedings of ICCD, pp.565-571, 1986.

[Chu87] Chu T.A., Synthesis of Self-timed VLSI Circuits from Graph-Theoretic Specifications. PhD thesis, M.I.T., MIT/LCS/TR-393, June 1987.

[Clar67] Clark W.A., Macromodular Computer Systems. AFIPS Conference Proceedings: 1967 Spring Joint Computer Conference, pp. 335-336. Academic Press, Atlantic City, NJ, 1967.

[Clar74] Clark W.A., Molnar C.E., Macromodular Computer Systems. Computers in Biomedical Research, editors Stacy R.W., Waxman B.D. chapter 3, pages 45-85. Academic Press, 1974.

[Coat93] Coates B., Davis A., Stevens K.S., Automatic Synthesis of Fast Compact Self-timed Control Circuits. Proceedings of the IFIP working conference on Asynchronous Design Methodologies, Manchester, England, 1993.

[Coat93a] Coates B., Davis A., Stevens K.S., The Post Office Experience: Designing a Large Asynchronous Chip. to appear in INTEGRATION, 1993.

[Cock87] Cockerell P., ARM Assembly Language Programming. MTC, England, 1987.

[Comp91] Chip Compiler V8R3. Compass Design Automation Inc., San Jose, U.S.A., Document No: VSD 40360, 1991.

[Comp91a] Layout Editor V8R3. Compass Design Automation Inc., San Jose, U.S.A., Document No: VSD 10414, 1991.

[Comp91b] Behavioral Modeling Langauage. Compass Design Automation Inc., San Jose, U.S.A., Document No: VSD 19160, 1991.

[Comp91c] Netlist Comparison. In Layout Verification, Compass Design Automation Inc., San Jose, U.S.A., Document No: VSD 50107, 1991.

Chapter 11 : Bibliography

[Akel91] Akella V., Gopalakrishnan G., Hopcp: A concurrent hardware description language. Technical Report UUCS-TR-91-021, Department of Computer Science, University of Utah, 1991.

[Asyn93] Asynchronous Online Bibliography, accessed via electronic mail addressed to 'async-bib@win.tue.nl'.

[ARM91a] Library Cell SPICE simulations for CMN12 process. ARM Ltd., Cambridge, England, Confidential, 1991.

[ARM91b] ARM6 Macrocell datasheet. ARM Ltd, Cambridge, England, September 1991.

[ARM92] ARM PIE User Guide. ARM Ltd, Cambridge, England, 1992.

[Beer92] Beerel P.A., Meng T.H-Y., Automatic Gate-Level Synthesis of Speed-Independent Circuits. Proc. Int'l. Conf. Computer-Aided Design. IEEE Computer Society Press, November, 1992.

[Brun89] Brunvand E., Sproull R.F., Translating concurrent programs into delay-insensitive circuits. In ICCAD-89, pp.262-265, IEEE, November, 1989.

[Brun91] Brunvand E., Translating Concurrent Communicating Programs into Asynchronous Circuits. PhD thesis, Carnegie Mellon University, 1991.

[Brun91a] Brunvand E., Starkey M., An Integrated Environment for the Design and Simulation of Self Timed Systems. VLSI 91, pp. 4a.2.1-4a.3.1. IFIP, August, 1991.

[Brun91b] Brunvand E., A Cell set for self-timed design using Actel FPGAs. Technical Report UUCS-91-013, University of Utah, 1991.

[Brun91c] Brunvand E. Implementing self-timed systems with FPGAs. In FPGAs, chapter 6.2, pp. 312-323, editors W.R. Moore and W.Luk, Abingdon EE&CS Books, 1991.

[Brun92] Brunvand E., Michell, N., Smith, K. A, Comparison of Self-Timed Design using FPGA, CMOS, and GaAs Technologies. 1992 IEEE International Conference on Computer Design: VLSI in Computers & Processors, pp. 76-80. October, 1992.

[Brun93] Brunvand E., The NSR Processor. 26th Hawaii Int. Conference on System Science (HICSS 1993), pp. 428-435. January, 1993.