ATMEL

Programmable Logic Devices

ATF15xx Family Device Fitter
Device Fitting Software

User Manual

Section 1

Introduction

This manual describes how to use the Atmel^{TM} ATF15xx device fitter for Atmel-WinCUPL, Prochip Designer, Protel design tools and legacy tools such as ABEL^{TM}, Atmel-Synario. The ATF15xx family of Complex Programmable Logic Devices [CPLDs] includes the ATF1500A, ATF1502xx, ATF1504xx, ATF1508xx and ATF1516xx devices. The fitter is required to fit designs into an Atmel CPLD and generate a JEDEC file. The macrocell (MC) and device features of this family will be introduced along with fitter installation requirements. The various fitter options and the fitting process are described in detail. Design examples are included to highlight the features of the ATF15xx family of devices and the various fitter command options.

In this manual, device fitter strategies for the ATF1500A differ slightly from the strategies available for the ATF15xx device fitters.

ATF15xx Family Architectural Features

The ATF15xx family includes the ATF1500A and ATF15xx devices (ATF1502xx, ATF1504xx, ATF1508xx and the ATF1516xx). In-system programming (ISP) capability and low voltage/low power offering of these devices is shown in Table 1.

The ATF1500A and the ATF1502xx are offered in 44-lead PLCC/TQFP packages. The ATF1504xx is offered in 44-lead PLCC/TQFP, 68-lead PLCC, 84-lead PLCC and 100-lead TQFP/PQFP packages. The ATF1508xx is offered in 84-lead PLCC, 100-lead TQFP/PQFP and 160-lead PQFP packages. Additional package options such as BGA will be offered for the ATF15xxSE and the ATF15xxAE family of devices.

Atmel Device	Number of Macrocells (MC)	ISP capability	Description
ATF1500A	32	No	5 V Standard Power Device
ATF1500AL	32	No	5 V Low Power Device
ATF1500ABV	32	No	2.7 V Standard Power Device
ATF1502AS, ATF1502SE ATF1504AS, ATF1504SE ATF1508AS, ATF1508SE ATF1516SE	32 64 128 256	Yes	5 V Standard Power Device
ATF1502ASL, ATF1502SEL ATF1504ASL, ATF1504SEL ATF1508ASL, ATF1508SEL ATF1516SEL	32 64 128 256	Yes	5 V Low Power Device
ATF1502ASV, ATF1502AE ATF1504ASV, ATF1504AE ATF1508ASV, ATF1508AE ATF1516AE	32 64 128 256	Yes	3.3 V Standard Power Device
ATF1502ASVL, ATF1502AEL ATF1504ASVL, ATF1504AEL ATF1508ASVL, ATF1508AEL ATF1516AEL	32 64 128 256	Yes	3.3 V Low Power device

Table 2-1. Atmel's ATF15xx Family of CPLDs

The macrocell architecture of the ATF1502xx, ATF1504xx, ATF1508xx and ATF1516xx is identical and differs slightly from that of the ATF1500A.

The ATF1500A (non-ISP device) is a globally connected part in that every pin can be routed to any other pin whereas the ATF1502AS (ISP capable) has a specific number of pins, feedback signals that can be routed to a Logic Array Blocks (LAB) via a Universal Interconnect Matrix [UIM]. Each LAB consists of 16 MCs.

Some highlights of the the ATF1500A device are:

- 32 I/O pins each connected to a macrocell [MC]
- 1 Global Clock or dedicated input pin
- 1 Global Reset or dedicated input pin

- 2 Global Output Enable or dedicated input pins
- All I/O pin and registered feedbacks connected to a Global Input Bus
- All Foldback Logic connected to a Regional Foldback Bus

All inputs connected to the Global Input Bus are available to *every* MC in the device. All inputs connected to the Foldback Bus are available to a group of 16 MCs. Figure 2-1 shows the block diagram of the device.

```
     (40)   2 INPUT/OE1                                        INPUT/CLK   43
     (38)  44 INPUT/OE2                                        INPUT/GCLR  1  (39)

     Corner Macrocell                                          Global Input/Feedback Bus
                                                               Corner Macrocell

(42)  4 I/O/Power Down       MC 1   EXP          EXP  MC 17         I/O  41
                                    BF           BF

     (43)   5                MC 2   EXP          EXP  MC 18         I/O  40
                                    BF           BF

     (44)   6                MC 3   EXP          EXP  MC 19         I/O  39
                                    BF           BF

      (1)   7                MC 4   EXP          EXP  MC 20         I/O  38
                                    BF           BF

      (2)   8                MC 5   EXP          EXP  MC 21         I/O  37
                                    BF           BF

      (3)   9                MC 6   EXP          EXP  MC 22         I/O  36
                                    BF           BF

      (5)  11                MC 7   EXP          EXP  MC 23         I/O  34
                                    BF           BF

      (6)  12                MC 8   EXP          EXP  MC 24         I/O  33
                                    BF           BF

     Corner Macrocell                                          Corner Macrocell

      (7)  13                MC 9   EXP          EXP  MC 25         I/O  32
                                    BF           BF

      (8)  14                MC 10  EXP          EXP  MC 26         I/O  31
                                    BF           BF

     (10)  16                MC 11  EXP          EXP  MC 27         I/O  29
                                    BF           BF

     (11)  17                MC 12  EXP          EXP  MC 28         I/O  28
                                    BF           BF

     (12)  18                MC 13  EXP          EXP  MC 29         I/O  27
                                    BF           BF

     (13)  19                MC 14  EXP          EXP  MC 30         I/O  26
                                    BF           BF

     (14)  20                MC 15  EXP          EXP  MC 31         I/O  25
                                    BF           BF

     (15)  21                MC 16  EXP          EXP  MC 32         I/O  24
                                    BF           BF

                              Regional      Regional
                              Foldback      Foldback
                               Bus            Bus
```

(Arrows connecting macrocells indicate direction and groupings of CASIN/CASOUT data flow)

Corner Macrocells that cannot borrow product terms from adjacent macrocell are: (4,13,32,41) for PLCC. (7,26,35,42) for TQFP

Figure 2-1. ATF1500A Block Diagram

7

2.1 Macrocell [MC] Features

Specific features of each MC include:

ATF1500A Macrocell	ATF15xx Macrocell
Global or Product Term controlled Output Enable originate from the Input Pins	Global [GOE0-GOE5]or Product Term controlled Output Enable originate from the Switch Matrix
Global or Product Term controlled Reset inputs	Global or Product Term controlled Reset inputs
Global or Product Term controlled Clock inputs	3 Global Clocks or Product term Clock input
Product-Term controlled Clock Enable and Preset inputs	Product-Term controlled Clock Enable and Preset inputs
5 product terms per MC	5 product terms per MC
Foldback and Cascade Logic	Foldback and Cascade logic
Independently configurable D/T/L Flip-flop	Independently configurable D/T/L Flip-flop.
Combinatorial Output with a Buried Register option	Combinatorial Output with a Buried Register option or Buried Combinatorial ouput with Registered Output.
Pin controlled or Automatic Power Down Control options	Pin controlled or Automatic Power Down Control options
Slew Rate Control on each output	Slew Rate Control on each output
Programmable Pin Keeper Circuits	Programmable Pin Keeper Circuits
The ATF1500A is a non-ISP device	4 JTAG port pins that allow for ISP
	Open Collector Outputs
	Fast Input from the I/O pin

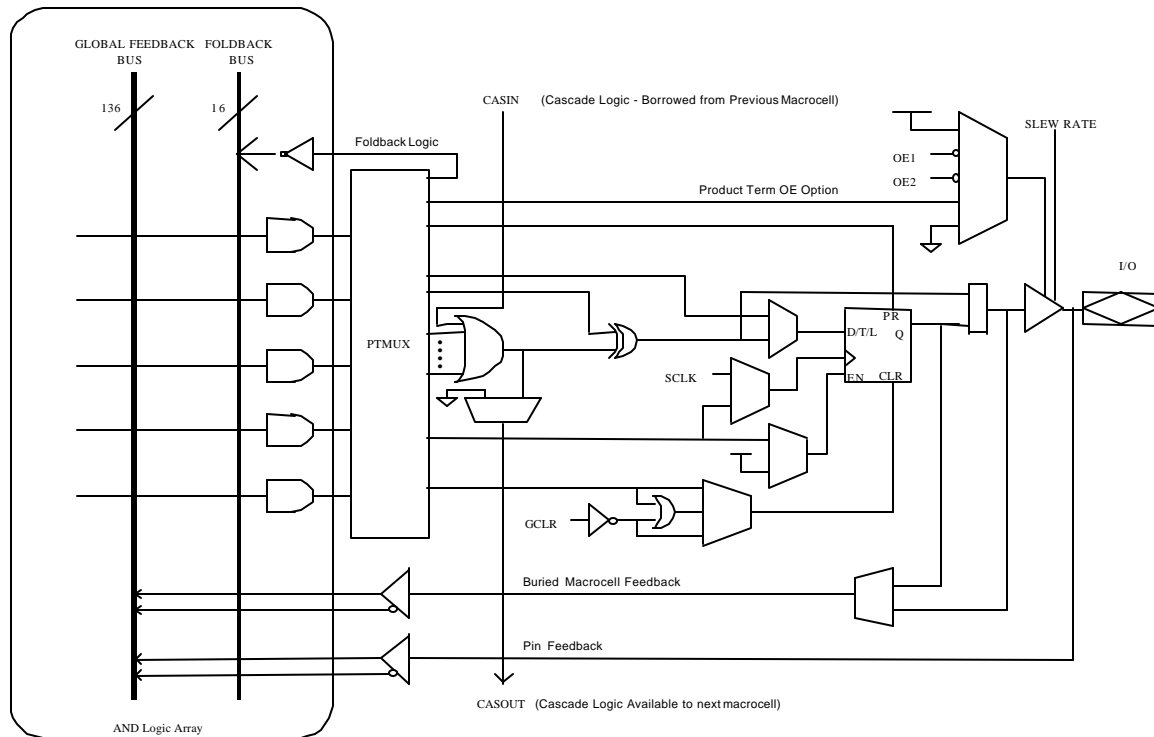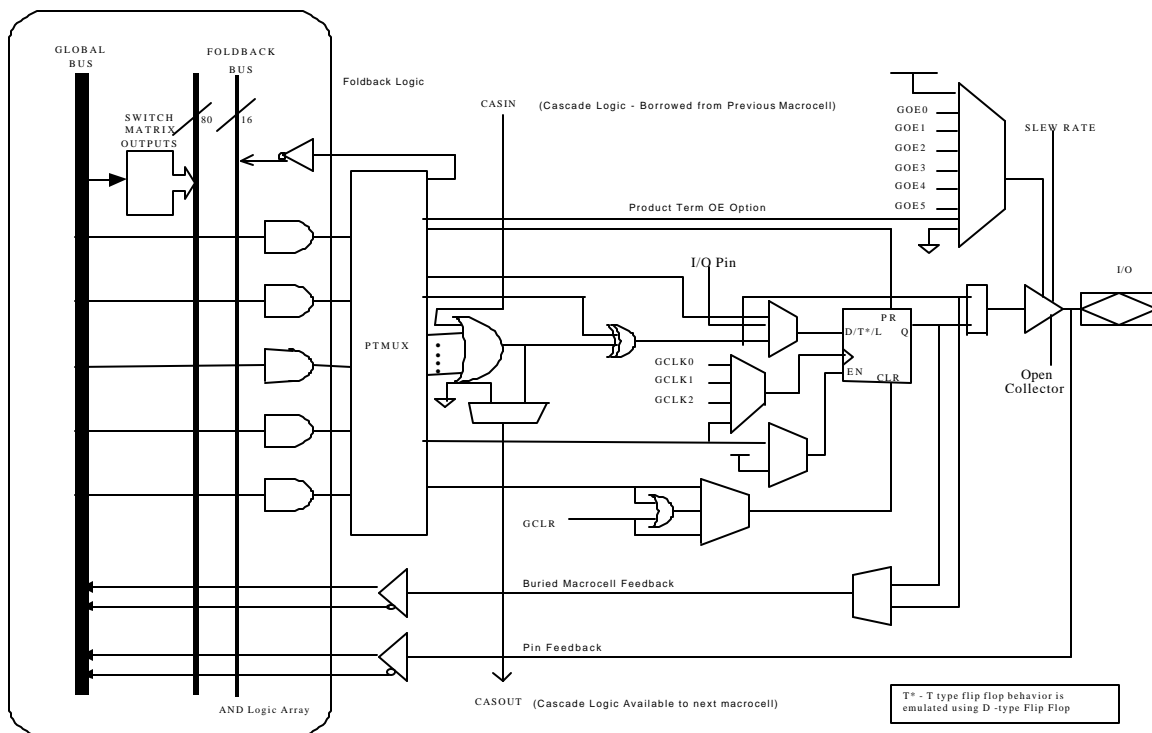Figure 2-2 shows the ATF1500A MC in more detail.

Figure 2-2 ATF1500A Macrocell

This MC is the same for ATF1502xx, ATF1504xx, ATF1508xx, ATF1516xx

Figure 2-3 ATF15xx Macrocell

Figure 2-3 shows the MC architecture for the ATF15xx MC.

2.2 Foldback Logic

Each ATF15xx/ATF1500A MC contains foldback logic nodes which allow complex functions requiring additional logic to be implemented. All foldback logic nodes are connected to the Foldback bus. The ATF15xx/ATF1500A contains two foldback busses each allocated to a group of 16 MCs. Foldback logic nodes for the first group containing macrocells MC1 through MC16 are connected to one of the foldback busses. The Foldback logic nodes for the second group of macrocells, MC17 through MC32, are connected to the other foldback bus. See Figure(s) 1 and 2 for more details.

2.3 Cascade Logic

Cascade logic allow logic to be borrowed from an adjacent MC. It is available for all macrocells, except the corner macrocells MC1, MC9, MC17 and MC25. This feature is useful for designs which require large product term fan-in such as state machines or high performance designs where speed is important. Each macrocell can borrow up to 5 product terms from an adjacent macrocell. Several macrocells can be cascaded together to provide up to 40 products terms for a single macrocell.

2.4 Power Down Control

All devices that belong to the ATF15xx Family have a Pin(s) {PD1, PD2}, which when asserted will power down the array and consume very low power. For example, the ATF1502AS is available in both standard and Low Power versions (ATF1502AS, ATF1502ASL respectively). The ATF1502AS has a Pin-Controlled power down pin option which, when enabled, allows the user to power down the device to a zero power mode where it draws a standby current in the microampere range. The ATF1502ASL low power device has

9

an Automatic power down feature. This feature powers down the device when no transitions occur on *any*.input, internal feedbacks or I/O pins. The ATF1502xx fitter has a command option to enable the Pin-Controlled power down feature. Please refer to Section 4.2.9 for more information.

2.5 Slew Rate Control

Each macrocell output on the ATF15xx has a programmable slew rate control bit. This bit can be used to to reduce system noise by slowing down the output that do not need to operate a maximum speed. All outputs default to the slower switching rate unless specified as fast switching in the ATF15xx fitter Slew Rate Control option. Please refer to Section 4.2.8 for more information on the specific strategy {**output_fast**}.

2.6 Logic Doubling

The term 'Logic_doubling' refers to the efficient , flexible CPLD architecture now available with second generation fitters and applies to all ATF15xx members except the ATF1500A which is a 100% connected CPLD. The ATF15xx Family has a number of features that address connectivity and reusability problems.
- More crosspoint Multiplexer (MUXs) available for input node fan-in
- Wider MUX channels into the Logic Array Blocks (LABs)
- Dual, independent feedback paths for each macrocell. The buried and pin-driver paths are split, so a register output may be buried while independently driving a combinatorial pin, or vice versa.
- Every MC may have separate Output Enable (product term controlled).
- Selectable Global Clock polarity, either falling or rising edge
- Global RESET can combine (OR) with a local product term

Using the ATF15xx Fitter

This section describes how to install and use the Atmel ATF15xx device fitter with PROCHIP, ABEL(ATMEL-SYNARIO) and CUPL PLD development software.

_____

3.1 INSTALLATION

The ATF15xx family of fitters are available for download from the Atmel website. {File Fit15xx.zip has the latest versions of Fitters that were released.} If you have any questions about obtaining these files please send an email to pld@atmel.com or call the Atmel PLD Applications Hotline at (408)436-4333.

3.1.1 PROCHIP DESIGNER

If you have already installed the Prochip Design tool, you can simply update the <Prochip>\PLDFIT Folder with a later version. Simply unzip the Fit15xx.zip file into this folder. Automatic updates to Prochip are also planned via the web.

3.1.2 ATMEL-WINCUPL

For customers who have previously installed Atmel-WinCupl Ver. 5.x, the FIT15xx.zip file can always be unzipped into the <Wincupl>\Wincupl\Fitters Folder. The current version of Atmel-Wincupl available for download off the website as a single file always contains the latest versions of the Device Fitters.

3.1.3 ALTIUM (PROTEL) TOOLS

The Design Explorer 99SE tool from ALTIUM (formerly called PROTEL) which is bundled in with the Prochip Design tool does not require a separate update, so long as the Fitters are being referenced from the Prochip tool.

If Design Explorer 99SE is installed as a separate tool (or if you have the full version of the tool from PROTEL), then the default location for the Fitters is in the <Programs>\Design Explorer 99 SE\library\pld folder.

3.1.4 ATMEL SYNARIO

If you have ATMEL SYNARIO V4.11 (has ABEL 6.x), the ATF15xx fitter software is included in the Atmel Device Kit for the appropriate tool. A Patch file {SNPCH411.exe} is available off the Atmel website. This is a single file install and will automatically copy the Fitters to the appropriate folders.

3.1.5 LEGACY VERSIONS OF ABEL (4.2x-5.x), CUPL (4.5x)

If you have Atmel-CUPL (V4.5b) you will need to install the fit1500.exe from the fit15xx.zip file. You can thus design with the ATF1500A device only. Similarly, Data I/O ABEL (V4.10 - 5.10) or Atmel-ABEL (4.2x-5.10) users will need to install the ATF1500 fitter. The older versions of ABEL (or stand alone Data I/O versions from the early nineties) do not have device support for the ATF15xx family, in particular the ATF1502xx or the ATF1504xx.

_____
_____

3.2 Device Support

All device packages as listed in the appropriate data sheet for an ATF15xx are supported. The Atmel ATF1500 fitter supports both the ATF1500A PLCC-44 and TQFP-44 packages. Specific device names (mnemonics) for ABEL(ATMEL-SYNARIO) and CUPL(applicable to WinCUPL/PROTEL) are listed in Section 3.2.2.

3.2.1 Legacy ABEL (DOS Versions)

The ATF1500A is supported in legacy versions of ABEL (V4.10 -5.1) by the device mnemonics listed below. The ATF1500 which was the first device introduced is no longer in production and the replacement part is the ATF1500A.

Device mnemonics are useful if the fitting process is run from the DOS prompt (ATMEL-SYNARIO).

3.2.2 ATMEL ATF15xx Device Names (mnemonics)

Table 3-1 lists the device mnemonics that must be specified when running the fitter manually from the DOS command line. The ATMEL WinCUPL device mnemonics listed are applicable to the header section of a CUPL source file only. However it is not important to remember the device menmonic when running ATMEL WinCUPL 5.2x. {Device names can be viewed by Package type from the Options...Devices Tab}

Atmel Synario also does not require the USER to specify a device mnemonic. Simply select the appropriate Atmel device from the list of ATMEL PLDs from the Project Sources window and proceed to generate the Fitter Report and JEDEC file used to program the device.

Select the In-system programming (ISP) device type from Table 3-1, when the JTAG port (4 pins) are to be exclusively used only for ISP operation.

The appropriate device mnemonic can also be specified when running the fitter from the DOS prompt.

Examples of running the Fitter in different Environments.

> ⇨ **FIT1502 filename.tt2 -dev p1502c44 -str JTAG=ON**
> Filename.tt2 is an intermediate PLA file created by the ABEL Compiler.

> ⇨ **FIT1502 filename.tt2 -cupl -dev p1502c44 -str JTAG=on -str preassign=keep**
> Filename.tt2 is an intermediate PLA file created by the ATMEL WinCUPL tool. Note that there is a slight difference between the ABEL PLA and CUPL PLA files.

> ⇨ **FIT1502 -i filename.edf -ifmt EDIF -lib aprim.lib -dev p1502c44 -str JTAG=off**
> If the input file is an EDIF file (EDIF version 2.0) created by a VHDL compiler (PEAKFPGA/PROCHIP)

> ⇨ **FIT1502 -c filename.cmd**
> The command file contains the list of all strategies and can be used if the command string is too long.

ATMEL DEVICE	ABEL Device Mnemonic (Device type when manually running fitter from the DOS prompt)	ATMEL WinCUPL Device Mnemonic (Applicable to Header section or for the Protel Tool)
ATF1500/L - PLCC44	p1500	f1500
ATF1500/L -TQFP44	p1500t	f1500t
ATF1500A/AL/ABV - PLCC	p1500a	f1500a
ATF1500A/AL/ABV -TQFP	p1500at	f1500at
ATF1502xx - PLCC44	p1502c44	f1502plcc44
ATF1502xx - ISP PLCC44	p1502c44 (Use -str JTAG=on)	f1502ispplcc44
ATF1502xx -TQFP44	p1502t44	f1502tqfp44
ATF1502xx -ISP TQFP44	p1502t44 (Use -str JTAG=on)	f1502isptqfp44
ATF1504xx - PLCC44	p1504c44	f1504plcc44
ATF1504xx - ISP PLCC44	p1504c44 (Use -str JTAG=on)	f1504ispplcc44
ATF1504xx -TQFP44	p1504t44	f1504tqfp44
ATF1504xx -ISP TQFP44	p1504t44 (Use -str JTAG=on)	f1504isptqfp44
ATF1504xx - PLCC68	p1504c68	f1504plcc68
ATF1504xx - ISP PLCC68	p1504c68 (Use -str JTAG=on)	f1504ispplcc68
ATF1504xx - PLCC84	p1504c84	f1504plcc84
ATF1504xx - ISP PLCC84	p1504c84 (Use -str JTAG=on)	f1504plcc84
ATF1504xx - TQFP100	p1504t100	f1504tqfp100
ATF1504xx - ISP TQFP100	p1504t100 (Use -str JTAG=on)	f1504isptqfp100
ATF1504xx - PQFP100	p1504q100	f1504pqfp100
ATF1504xx - ISP PQFP100	p1504q100 (Use -str JTAG=on)	f1504isppqfp100
ATF1508xx - PLCC84	p1508c84	f1508plcc84
ATF1508xx -ISP PLCC84	p1508c84 (Use -str JTAG=on)	f1508ispplcc84
ATF1508xx -PQFP100	p1508q100	f1508qfp100
ATF1508xx - ISP PQFP100	p1508q100 (Use -str JTAG=on)	f1508ispqfp100
ATF1508xx -TQFP100	p1508t100	f1508tqfp100
ATF1508xx - ISP TQFP100	p1508t100 (Use -str JTAG=on)	f1508isptqfp100
ATF1508xx -PQFP160	p1508q160	f1508pqfp160
ATF1508xx-ISP PQFP160	p1508q160 (Use -str JTAG=on)	f1508isppqfp160

Table 3-1. ATF15xx Device Mnemonics

3.3 Pin/Node Maps for the ATF15XX Family.

The pin and node assignments for **ATF1500A** in PLCC and TQFP package are shown in Table 3-2. This table also shows the node assignment for the Foldback logic and Buried registers (feedback node) available in each macrocell [MC]. These node numbers are the same for ABEL, CUPL and VHDL.

MC	Foldback Node Number	Feedback Node Number	Pinout 44-Pin PLCC	Pinout 44-Pin TQFP
1	45	77	4	42
2	46	78	5	43
3	47	79	6	44
4	48	80	7	1
5	49	81	8	2
6	50	82	9	3
7	51	83	11	5
8	52	84	12	6
9	53	85	13	7
10	54	86	14	8
11	55	87	16	10
12	56	88	17	11
13	57	89	18	12
14	58	90	19	13
15	59	91	20	14
16	60	92	21	15
17	61	93	41	35
18	62	94	40	34
19	63	95	39	33
20	64	96	38	32
21	65	97	37	31
22	66	98	36	30
23	67	99	34	28
24	68	100	33	27
25	69	101	32	26
26	70	102	31	25
27	71	103	29	23
28	72	104	28	22
29	73	105	27	21
30	74	106	26	20
31	75	107	25	19
32	76	108	24	18

Table 3-2. ATF1500A Pin/Node List

Table 3-3 shows the Pin and Node assignments for the **ATF1502xx** device in PLCC and TQFP package. Feedback and Foldback nodes numbers are listed though it is preferred that the fitter be allowed to arrive at the best possible fit without preassigning any Pinnode numbers.

MC	LAB	Feedback Node Number	Foldback Node Number	Signal Name	Pinout 44-Pin PLCC	Pinout 44-Pin TQFP
INPUT		none	none	(OE2/GCLK2)	2	40
INPUT		none	none	(GCLK1)	43	37
INPUT		none	none	(GCLR)	1	39
INPUT		none	none	(OE1)	44	38
1	A	601	301		4	42
2		602	302		5	43
3		603	303		6	44
4		604	304		7 (TDI)	1(TDI)
5		605	305		8	2
6		606	306		9	3
7		607	307		11	5
8		No Casout 608	308		12	6
9		609	309		13(TMS)	7(TMS)
10		610	310		14	8
11		611	311		16	10
12		612	312		17	11
13		613	313		18	12
14		614	314		19	13
15		615	315		20	14
16		No Casout 616	316		21	15
17	B	617	317		41	35
18		618	318		40	34
19		619	319		39	33
20		620	320		38(TDO)	32(TDO)
21		621	321		37	31
22		622	322		36	30
23		623	323		34	28
24		No Casout 624	324		33	27
25		625	325		32(TCK)	26(TCK)
26		626	326		31	25
27		627	327		29	23
28		628	328		28	22
29		629	329		27	21
30		630	330		26	20
31		631	331		25	19
32		No Casout 632	332		24	18

Table 3-3. ATF1502xx Pin/Node List

Table 3-4 lists the Pin and Node numbers for the **ATF1504xx** device in various packages. It is preferred that Foldback node numbers as well as Buried Feedback node numbers are not preassigned in the Source files so that the Fitter can fit the Design in a manner that allows for optimum use of resources within the Device.

MC	LAB	Feedback Node Number	Foldback Node Number	Signal Name	Pinout for ATF1504xx					
					44-Pin PLCC	44-Pin TQFP	68-Pin PLCC	84-Pin PLCC	100-Pin PQFP	100-Pin TQFP
INPUT		none	none	(OE2/GCLK2)	2	40	2	2	92	90
INPUT		none	none	(GCLK1)	43	37	67	83	89	87
INPUT		none	none	(GCLR)	1	39	1	1	91	89
INPUT		none	none	(OE1)	44	38	68	84	90	88
1	A	601	301		12	6	18	22	16	14
2		602	302		-	-	-	21	15	13
3		603	303		11	5	17	20	14	12
4		604	304		9	3	15	18	12	10
5		605	305		8	2	14	17	11	9
6		606	306		-	-	13	16	10	8
7		607	307		-	-	-	15	8	6
8		No Casout 608	308		7	1	12	14	6	4
9		609	309		-	-	10	12	4	100
10		610	310		-	-	-	11	3	99
11		611	311		6	44	9	10	100	98
12		612	312		-	-	8	9	99	97
13		613	313		-	-	7	8	98	96
14		614	314		5	43	5	6	96	94
15		615	315		-	-	-	5	95	93
16		No Casout 616	316		4	42	4	4	94	92
17	B	617	317		21	15	33	41	39	37
18		618	318		-	-	-	40	38	36
19		619	319		20	14	32	39	37	35
20		620	320		19	13	30	37	35	33
21		621	321		18	12	29	36	34	32
22		622	322		-	-	28	35	33	31
23		623	323		-	-	-	34	32	30
24		No Casout 624	324		17	11	27	33	31	29
25		625	325		16	10	25	31	27	25
26		626	326		-	-	-	30	25	23
27		627	327		-	-	24	29	23	21
28		628	328		-	-	23	28	22	20
29		629	329		-	-	22	27	21	19
30		630	330		14	8	20	25	19	17
31		631	331		-	-	-	24	18	16
32		No Casout 632	332		13	7	19	23	17	15
33	C	633	333		24	18	36	44	42	40
34		634	334		-	-	-	45	43	41
35		635	335		25	19	37	46	44	42
36		636	336		26	20	39	48	46	44
37		637	337		27	21	40	49	47	45
38		638	338		-	-	41	50	48	46
39		639	339		-	-	-	51	49	47
40		No Casout 640	340		28	22	42	52	50	48
41		641	341		29	23	44	54	54	52
42		642	342		-	-	-	55	56	54
43		643	343		-	-	45	56	58	56
44		644	344		-	-	46	57	59	57
45		645	345		-	-	47	58	60	58
46		646	346		31	25	49	60	62	60
47		647	347		-	-	-	61	63	61
48		No Casout 648	348		32	26	50	62	64	62

ATF1504xx Pin/Node Worksheet Cont'd										
MC	LAB	Feedback Node Number	Foldback Node Number	Signal Name	Pinout					
					44-Pin PLCC	44-Pin TQFP	68-Pin PLCC	84-Pin PLCC	100-Pin PQFP	100-Pin TQFP
49	D	649	349		33	27	51	63	65	63
50		650	350		-	-	-	64	66	64
51		651	351		34	28	52	65	67	65
52		652	352		36	30	54	67	69	67
53		653	353		37	31	55	68	70	68
54		654	354		-	-	56	69	71	69
55		655	355		-	-	-	70	73	71
56		No Casout 656	356		38	32	57	71	75	73
57		657	357		39	33	59	73	77	75
58		658	358		-	-	-	74	78	76
59		659	359		-	-	60	75	81	79
60		660	360		-	-	61	76	82	80
61		661	361		-	-	62	77	83	81
62		662	362		40	34	64	79	85	83
63		663	363		-	-	-	80	86	84
64		No Casout 664	364		41	35	65	81	87	85

Table 3-4. ATF1504xx Pin/Node List

Table 3-5 lists the Pin and Node numbers for the **ATF1508xx** device in various packages. It is preferred that Foldback node numbers as well as Buried Feedback node numbers are not preassigned in the Source files.

MC	LAB	Feedback Node Number	Foldback Node Number	Signal Name	Pinout for ATF1508xx			
					84-Pin PLCC	100-Pin PQFP	100-Pin TQFP	160-Pin PQFP
INPUT		none	none	(OE2/GCLK2)	2	92	90	142
INPUT		none	none	(GCLK1)	83	89	87	139
INPUT		none	none	(GCLR)	1	91	89	141
INPUT		none	none	(OE1)	84	90	88	140
1	A	601	301		--	4	2	160
2		602	302		--	--	--	--
3		603	303		12	3	1	159
4		604	304		--	--	--	158
5		605	305		11	2	100	153
6		606	306		10	1	99	152
7		607	307		--	--	--	--
8		No Casout 608	308		9	100	98	151
9		609	309		--	99	97	150
10		610	310		--	--	--	--
11		611	311		8	98	96	149
12		612	312		--	--	-	147
13		613	313		6	96	94	146
14		614	314		5	95	93	145
15		615	315		--	--		--
16		No Casout 616	316		4	94	92	144
17	B	617	317		22	16	14	21
18		618	318		--	--		--
19		619	319		21	15	13	20
20		620	320		--	--		19
21		621	321		20	14	12	18
22		622	322		--	12	10	16
23		623	323		--	--		--

MC	LAB	Feedback Node Number	Foldback Node Number	Signal Name	84-Pin	100-Pin PQFP	100-pin TQFP	160-Pin PQFP
24		No Casout 624	324		18	11	9	15
25		625	325		17	10	8	14

ATF1508xx Pin/Node Worksheet Cont'd								
MC	**LAB**	**Feedback Node Number**	**Foldback Node Number**	**Signal Name**	**Pinout**			
					84-Pin	**100-Pin PQFP**	**100-pin TQFP**	**160-Pin PQFP**
26	B	626	326		--	--		--
27		627	327		16	9	7	13
28		628	328		--	--		12
29		629	329		15	8	6	11
30		630	330		--	7	5	10
31		631	331		--	--		--
32		No Casout 632	332		14	6	4	9
33	C	633	333		--	27	25	41
34		634	334		--	--		--
35		635	335		31	26	24	33
36		636	336		--	--		32
37		637	337		30	25	23	31
38		638	338		29	24	22	30
39		639	339		--	--		--
40		No Casout 640	340		28	23	21	29
41		641	341		--	22	20	28
42		642	342		--	--		--
43		643	343		27	21	19	27
44		644	344		--	--		25
45		645	345		25	19	17	24
46		646	346		24	18	16	23
47		647	347		--	--		--
48		No Casout 648	348		23	17	15	22
49	D	649	349		41	39	37	59
50		650	350		--	--		--
51		651	351		40	38	36	58
52		652	352		--	--		57
53		653	353		39	37	35	56
54		654	354		--	35	33	54
55		655	355		--	--		--
56		No Casout 656	356		37	34	32	53
57		657	357		36	33	31	52
58		658	358		--	--		--
59		659	359		35	32	30	51
60		660	360		--	--		50
61		661	361		34	31	29	49
62		662	362		--	30	28	48
63		663	363		--	--		--
64		No Casout 664	364		33	29	27	43
128	H	728	428		81	87	85	137
127		727	427		--	--		--
126		726	426		80	86	84	136
125		725	425		79	85	83	135
124		724	424		--	--		134
123		723	423		77	83	81	132
122		722	422		--	--		--
121		No Casout 721	421		--	82	80	131
120		720	420		76	81	79	130
119		719	419		--	--		--
118		718	418		75	80	78	129
117		717	417		74	79	77	128
116		716	416		--	--		123

MC	LAB	Feedback Node Number	Foldback Node Number	Signal Name	84-Pin	100-Pin PQFP	100-pin TQFP	160-Pin PQFP
115		715	415		73	78	76	122
114		714	414		--	--		--
113		No Casout 713	413		--	77	75	121
112	G	712	412		71	75	73	112
111		711	411		--	--		--
110		710	410		--	74	72	111

ATF1508xx Pin/Node Worksheet Cont'd

MC	LAB	Feedback Node Number	Foldback Node Number	Signal Name	Pinout			
					84-Pin	100-Pin PQFP	100-pin TQFP	160-Pin PQFP
109	G	709	409		70	73	71	110
108		708	408		--	--		109
107		707	407		69	72	70	108
106		706	406		--	--		--
105		No Casout 705	405		68	71	69	107
104		704	404		67	70	68	106
103		703	403		--	--		--
102		702	402		--	69	67	105
101		701	401		65	67	65	103
100		700	400		--	--		102
99		699	399		64	66	64	101
98		698	398		--	--		--
97		No Casout 697	397		63	65	63	100
96	F	696	396		62	64	62	99
95		695	395		--	--		--
94		694	394		61	63	61	98
93		693	393		60	62	60	97
92		692	392		--	--		96
91		691	391		58	60	58	94
90		690	390		--	--		--
89		No Casout 689	389		--	59	57	93
88		688	388		57	58	56	92
87		687	387		--	--		--
86		686	386		56	57	55	91
85		685	385		55	56	54	90
84		684	384		--	--		89
83		683	383		54	55	53	88
82		682	382		--	--		--
81		No Casout 681	381		--	54	52	80
80	E	680	380		52	52	50	78
79		679	379		--	--		--
78		678	378		--	51	49	73
77		677	377		51	50	48	72
76		676	376		--	--		71
75		675	375		50	49	47	70
74		674	374		--	--		--
73		No Casout 673	373		49	48	46	69
72		672	372		48	47	45	68
71		671	371		--	--		--
70		670	370		--	46	44	67
69		669	369		46	44	42	65
68		668	368		--	--		64
67		667	367		45	43	41	63
66		666	366		--	--		--
65		665	365		44	42	40	62

Table 3-5. ATF1508xx Pin/Node List

3.4 Operation

The ATF15xx family of fitters have been developed to run automatically in PROCHIP, ALTIUM (PROTEL) DESIGN EXPLORER 99SE Tool, ATMEL-SYNARIO V4.11 and ATMEL WinCUPL. Users can specify the device type in the source/Project file and the appropriate tool compiler will automatically execute the ATF15xx fitter software. You can customize fitter operation for ABEL or CUPL designs by using the DOS command line, PROPERTY statements or macros in the source file. These methods will be discussed in more detail later in this manual.

3.4.1 VHDL Designs (Prochip environment)

For VHDL Designs, you can compile the Source files using PROCHIP tools into a Netlist (EDIF 2.0). Once the EDIF file is generated, you can use the GUI from Prochip to invoke the **Device Fitter** properties window and select the *Global Device Parameters* or *Pin/Node options* to specify individual pin or MC properties. Users can click on the RUN FITTER button as shown in the Figure 3-1 to generate the Fitter Report File (.FIT) and the .JED (JEDEC programming file)
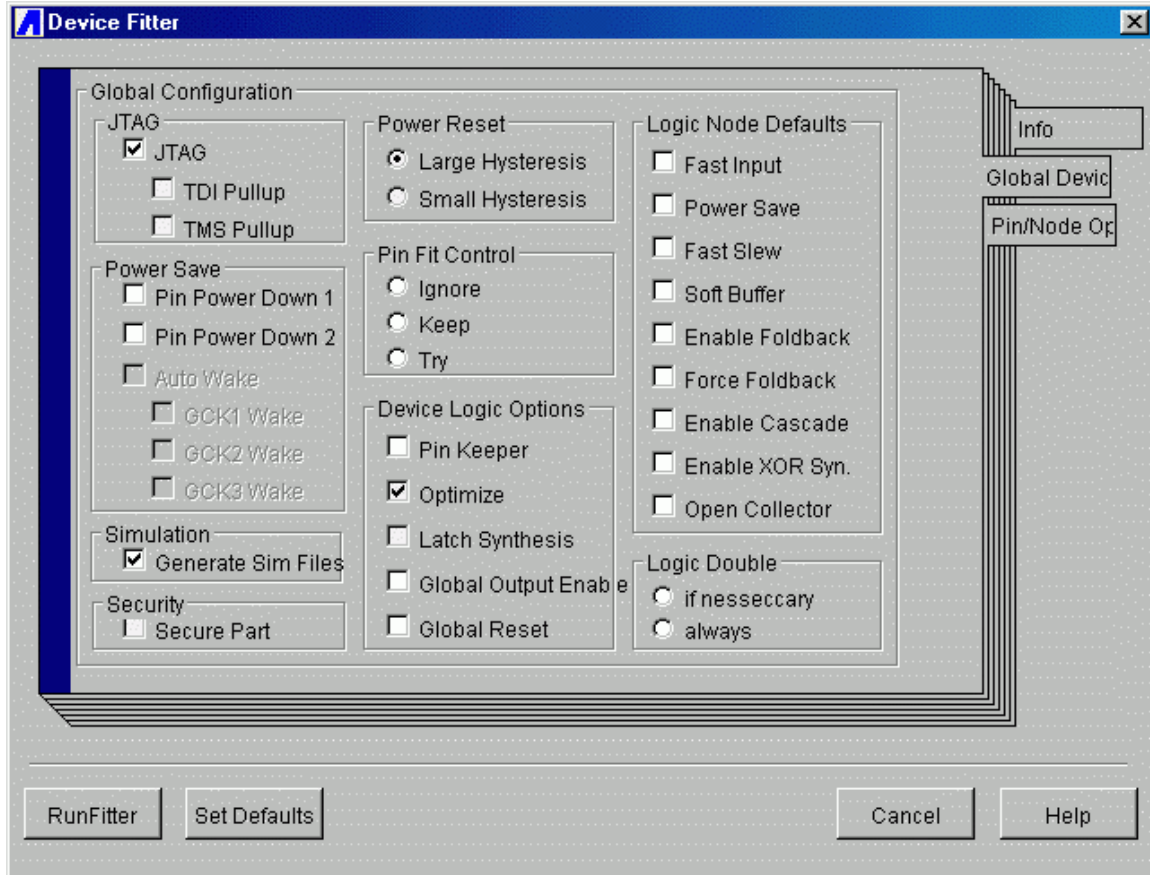

Figure 3-1

3.4.2 VHDL Designs (ATMEL-Synario environment)

In ATMEL-SYNARIO, Source vhdl (.vhd) files can be synthesized into PLA format (.tt2) files. The target device can be selected from a list of ATMEL devices. Alternatively, Fitters can also be run from the DOS prompt by clicking on the 'CTRL"-"ALT"-"D" keys simultaneously. Section 3.2.2 describes the syntax/command.

3.4.3 ABEL Designs (ATMEL-Synario and Legacy ABEL 5.1x)

In ATMEL-SYNARIO, Source ABEL (.abl) files can be synthesized into PLA format (.tt2) files. The target device can be selected from a list of ATMEL devices. Alternatively, Fitters can also be run from the DOS prompt by clicking on the 'CTRL"-"ALT"-"D" keys simultaneously. Section 3.2.2 describes the syntax/command.

In the DOS ABEL (Legacy ABEL 5.1x) environment if you are targetting the ATF1500A, enter the target device type by selecting the
Fit Options..
command from the ABEL SmartPart Menu. Enter the *p1500*a or *p1500at* device type and press *F5* key to exit.
Then select the
Fit
command from the SmartPart Menu to run the fitter and create the JEDEC file.

An alternate way to target the ATF1500 device is to include the appropriate ABEL device type shown in Table 3-1 in your ABEL design file by using the *Device* Statement. Then select the *Fit* Command above from the SmartPart menu to run the fitter.

☞ *Windows ABEL (ABEL 6.x) or ATMEL-SYNARIO users should select the ATF1500A device in your project and execute the* **Create Jedec File** *Process. The fitter will be executed automatically*.

3.4.4 CUPL/Schematic Designs (Prochip environment)

For CUPL/Schematic Designs, you can compile the Source files using PROCHIP tools. All Schematic files are first translated into a CUPL (.pld) file using the "virtual" device type. This file is then compiled into a PLA (.tt2) file. Once the PLA file is generated, you can use the GUI from Prochip to invoke the **Device Fitter** properties window and select the Global Parameters or Advanced options to specify individual pin or MC properties. Users can click on the RUN FITTER button to generate the Fitter Report File (.FIT) and the .JED (JEDEC programming file)

CUPL or Schematic Designs can alternatively be directly compiled using **DESIGN EXPLORER 99SE TOOL from ALTIUM (formerly known as PROTEL).**

3.4.5 CUPL (ATMEL- WinCUPL)

ATMEL-WINCUPL is the windows version of CUPL and is specific to the ATMEL device library only. All Atmel EPLDs are supported. Users can either specify the device mnemonic in the Header section as shown in Figure 3-2.

```
Name            Reglatch;
Partno          ;
Revision        01;
Date            8/11/95;
Designer        PLD Expert;
Company         Atmel Corp.;
Location        None;
Assembly        None;
Device          F1500a; /*Device mnemonic is specified in the Header section*/

/* Example showing register and latch usage */
```

Figure 3-2

For a list of devices, users can reference the **Help.Atmel INFO** button from within ATMEL-WinCUPL. Users are not required to remember the device mnemonic and can alternatively select the appropriate Atmel Device and then proceed to Compile. As shown in Figure 3-3, users can click on the **OPTIONS** button and then select the **Devices** option
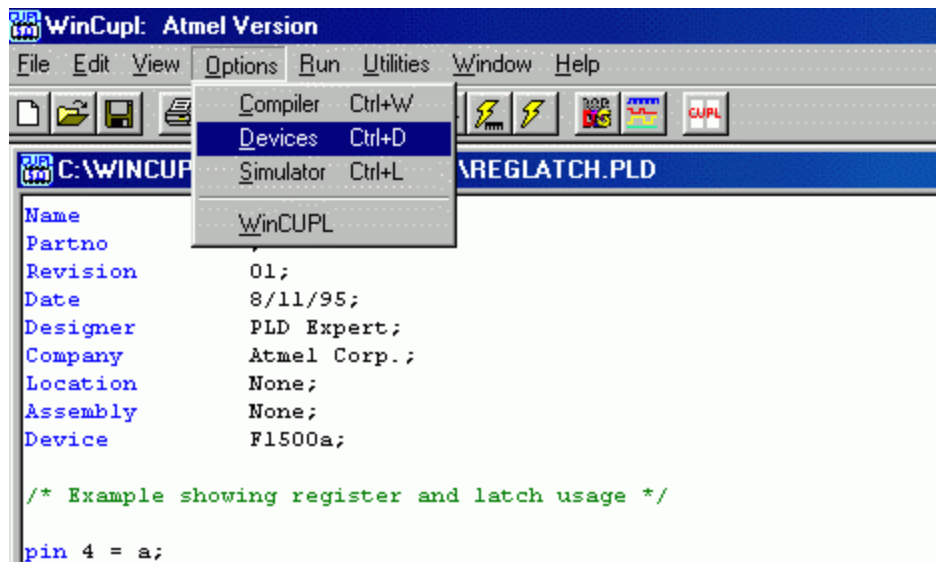

Figure 3-3

This brings up the GUI as shown in Figure 3-4. All ATMEL EPLDs are classified by package type. Please **de-select (Uncheck)** the **Device in File** option so that the WinCupl Compiler will only use the Device type selected in this GUI for the compilation process.
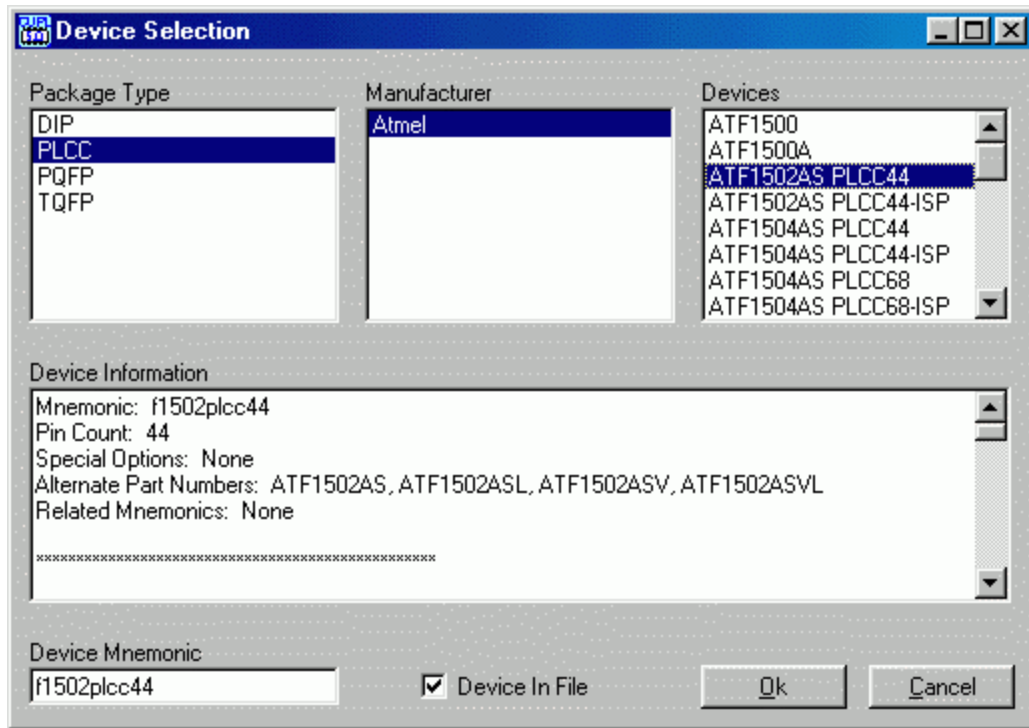
Figure 3-4

3.4.6 CUPL and MCUPL (Legacy versions of CUPL)

The following table lists the flow applicable to the old ATF1500/A devices. For new designs, we recommend that customers always use ATMEL-WinCUPL.

MCUPL (Legacy CUPL V4.x) Flow	CUPL (Legacy CUPL4.7x)
To use the ATF1500 fitter in MCUPL: select the, **<D>evice selection** command from the main menu to choose the appropriate ATF1500 CUPL device type, if it is not already included in your source file. Then Select, **<C>ompile Design File** Use F3 to change the DOS subdirectory, if necessary. Select the file and press *<return>*. An output options menu will appear. Select at least the **<J>edec File Output** option to run the fitter. Press the **F5** key to start the compilation process.	You may compile your design in CUPL from the DOS command line. Type, **CUPL -J{option list} <filename> <return>** at the command line. You must specify at least the *-J* option. Additional options can be specified. To view these options type, **CUPL <return>** For example, **CUPL -jxf c:\fit1500\project\project.pld** will instruct CUPL to compile the *project.pld* file located in the c*:\fit1500\project* directory. The -x and -f options specify expanded product terms and a fuse map listing to be produced by CUPL.

3.4.7 Command Line

The ATF15xx fitter can also be executed from the DOS command line. The command line is shown below for ATF1500A and ATF1502AS. A detailed description of each option will be discussed in the section 4.0 of this manual.

fit1500 [-i] *plafil*e**[.tt2] {***options***}** **Options:** **-help** **-o** *output_file_name* **(for *.tt3 and *.jed)** **-log** *report_file* **(*.doc)** **-err** *error_file* **(*.err)** **-device (***p1500/ p1500t/ f1500/ f1500t***)** **-preassign** *TRY|Keep|IGNore* **(pin preassignment options)** **-silent (no message on screen)** **-h2 (advanced help option)**	**fit1502 [-i] plafile[.tt2/edif]** **Options:** **-help** **-o output_file_name (for *.tt3 and *.jed)** **-device (or -dev) device_name (p1502c44/p1502t44)** **-module module_name** **-preassign TRY|keep|ignore (pin preassignment options)** **-silent (no message on screen)** **-h2 (advanced help option)**

Fit15xx Strategies

The ATF15xx fitter has both basic and advanced strategies (options) that allow you to customize the operation of the fitter to meet your design requirements. Both sets of options are discussed below.

4.1 Basic Strategies

4.1.1 Specifying Input Filename [-i]

This option allows you to instruct the fitter to read the input file as a different input filename or from a different directory than the default directory. The fitter defaults to reading a [.tt2/edif] input file conforming to the OPEN-ABEL PLA(or CUPL PLA) format from the working directory. This file is normally created by ABEL or CUPL compiler as part of the design compilation process. For example, the command

fit1500 design

executes the fitter from the command line and reads the *design.tt2* PLA file from the working directory. The command

fit1500 -i c:\wincupl\project\project1.tt2 -cupl

executes the fitter from the command line and reads the *project1.tt2* file from the *c:\wincupl\project* directory.

fit15xx -i c:\Prochip\Designs\vhdl\shift.edf -ifmt EDIF -lib aprim.lib -dev p1502c44 -str JTAG=ON

executes the fitter from the command line and reads the *shift.edf* file from the *c:\Prochip\Designs\vhdl* directory.

4.1.2 Specifying Output Filenames [-o]

This option allows you to specify different output filenames for the post-fit PLA and JEDEC files created by the fitter or save these files in a different directory than default directory. The fitters defaults by creating a post-fit PLA file, [.tt3] file and a JEDEC [.jed] file and saving these files in your working directory. For example the command,

fit1502 design -o c:\fit1502\project\project1.pla

The post fit PLA file is saved as *project1.pla* and the JEDEC file as *project1.jed* to the *c:\fit1502\project* subdirectory.

4.1.3 Fitter Report Filename [-log]

Use this option to save the fitter's report file either as a different filename or in different directory than the default directory. The fitter defaults by creating the report file as a **[.fit]** file and saving it in the working directory. The report file is an output file automatically created by the fitter which shows the results of the fitting process. The contents of this file and how they can be interpreted will be discussed in more detail in section 6.0 of this manual. Using the command,

fit1502 design -log c:\fit1502\project\project1.rep

The fitter save the report file as *project1.rep* in the *c:\fit1502\project* subdirectory.

4.1.4 Error File [-err]

Use this option to save the fitter's error file either as a different filename or in different directory than the default directory. The fitter defaults by creating the error file as a [.err] file and saving it in the working directory. The error file is created to save any errors or warnings encountered in the fitting process.

fit1500 design -err c:\fit1500\project\project1.err

The fitter error file is saved as *project1.err* to the *c:\fit1500\project* subdirectory.

4.1.5 Specify Device [-device]

If the original ABEL or CUPL source file does not include the device type, then you must use the device option to specify which ATF15xx device you want the fitter to use. Lists of allowable device types for both ABEL and CUPL are in Section 3.2 of this manual. This option is also useful if you want the fitter the re-target the design to a different ATF15xx device type. For example, if the original source file uses the PLCC device type, you can use the *device* command to override this device type to select the TQFP device type.

fit1500 design -device f1500t -cupl (Used for CUPL Designs)
fit1500 design -device p1500t (Used for ABEL Designs)
fit1502 -i shift.edf -ifmt EDIF -lib aprim.lib -dev p1502c44 -str JTAG=ON (Used for Prochip flow)

4.1.6 Preassign Pins [-pre TRY|Keep|Ignore]

This option specifies how the ATF15xx fitter should treat any pin or node assignments made in the VHDL, ABEL or CUPL source file. VHDL designs will generate a pinfile (.pin) through the Prochip flow. There are three preassignment options available for the ATF15xx fitter. These are *Keep, Try* and *Ignore*. The fitter defaults to using the TRY option if no preassignment option is specified.

4.1.6.1 Keep Option

This option instructs the fitter to keep all pin and node assignment specified in the source file during the fitting process. Unassigned pin and nodes in the design are assigned automatically by the fitter during the fitting process. This process will fail if the user-specified pin and node pre-assignments result in a design that does not fit into the ATF1502AS.

fit1502 design.tt2 -cupl -dev p1502c44 -pre Keep

The fitter will keep the original pin and node assignment when attempting to fit the *design.tt2* file.

4.1.6.2 Try Option

This option instructs the fitter to use the *Keep* option first to fit the design. If the design fails to fit then the *Ignore* option is used. Unassigned pin and nodes are assigned automatically by the fitter. This fitter will default to this option if no preassignment options are specified in the design file or command line.

4.1.6.3 Ignore Option

This option instructs the fitter to ignore any pin and node preassignment in the design file and assigns all pins and nodes.

fit1504 design.tt2 -cupl -dev p1504c44 -pre Ignore

The fitter ignores any pin or node assignments when attempting to fit the *design.tt2* file.

Please refer to section 5.1 for more information about these options.

4.1.7 Silent Option [-silent]
The option prevents the result of the fitting process from being displayed on the console.

fit1500 design -silent

4.1.8 HELP Option [-help]

fit1500 -help

Displays the basic fitting options that are available for the ATF1500A device. This option is not applicable to other ATF15xx devices

_____
___

4.2 Advanced Options and Strategies [-h2]

This option will display all of the advanced fitting option and strategies available to fit a design into the ATF15xx. These options allow you to customize the fitting process to help a design to fit efficiently to optimize it for performance. Each option is discussed below.

fit1502 -h2

Displays the advanced fitting options and strategies that are available for the ATF1502xx devices.

4.2.1 Optimize [-str optimize ON | off]
This strategy allows the user to control all of the fitter's logic optimization strategies for the ATF15xx. These strategies include: Cascade and Foldback logic, XOR Synthesis and the Node Collapsing feature.This strategy defaults to being set *on*.

fit1502 design.edf -ifmt EDIF -lib aprim.lib -dev p1502c44 -str jtag=on -str optimize=off

Will disable the logic optimization strategy and fit the design exactly as specified in the source file.

4.2.2 XOR Synthesis [-str xor_synthesis ON| Off = signal_name,...signal_name]

This strategy allows the user to control the use of the hardware XOR gate in the ATF15xx macrocell for synthesizing design logic. This option can be enabled for either all output signals or for individual signals. These signals can represent either pin or nodes in the design. The hardware XOR gate in the ATF15xx is useful for reducing product term count for implementing arithmetic logic and comparison functions, or any other logic functions that require an XOR. The hardware XOR gate will be used when this strategy is on and if the sythesized logic using the XOR gate results in fewer product term than the unsynthesized logic. This strategy defaults to being set *on*

fit1500 design -str xor_synthesis = a,b

Enables the XOR Synthesis feature for signals named *a* and *b* respectively. All remaining signals will have the XOR_synthesis feature turned *off.*

fit1504 design.tt2 -cupl -dev p1504t44 -str xor_sythesis on

Enables the XOR Synthesis feature for all macrocells in theATF1504xx.

fit1500 design -str xor_synthesis off

Disables the XOR Synthesis feature for all macrocells.

4.2.3 Cascade Logic [-str cascade_logic ON| Off = signal_name,..., signal_name]

This strategy allows the user to control whether the cascade logic feature in the ATF15xx macrocell is used for implementing logic functions. This option can be enabled for either all output signals or for individual signals that are listed. These signals can represent either pins or nodes in the design. The Cascade logic feature allows the user to borrow product terms from an adjacent macrocell with a small additional delay. This feature is useful for implementing high product term fan-in designs such as state machines or for optimizing a design for speed. This startegy defaults to being set *on*.

fit1502 design.tt2 -cupl dev p1502c44 -str cascade_logic = a,b

Enables the cascade logic feature for signals named *a* and *b* respectively in the source file. All other signals will have the Cascade Logic feature turned *off*.

fit1502 design.tt2 -cupl -dev p1502c44 -str cascade_logic on

Enables the Cascade logic feature for all macrocells in the device.

fit1502 design.tt2 -cupl -dev p1502c44 -str cascade_logic off

Disables the Cascade logic feature for all macrocells in the ATF1502 device.

4.2.4 Foldback Logic [-str foldback_logic ON| Off = signal_name,...,signal_name]

This strategy allows the user to control whether foldback logic nodes are used in the ATF15xx macrocell. This option can be enabled for either all signals or for individual signals. These signals represent nodes in the design. This feature is useful for fitting additional logic in the macrocell for design upgrades or modifications. This strategy defaults to being set *on*.

fit1500 design -strategy foldback_logic = a,b

Enables the Foldback logic nodes for signals named *a* and *b* respectively. All other signals will have the foldback logic feature turned *off*.

fit1502 design.tt2 -cupl -dev p1502c44 -str foldback_logic on

Enables the foldback logic nodes for all macrocells in the ATF1502.

fit1500 design.tt2 -cupl -dev p1502c44 -str foldback_logic off

Disables the foldback logic nodes for all macrocells.

4.2.5 Foldback Logic Compatibility [-strategy expander = node_name,....,node_name]

This strategy allows the user to define sharable expander logic equations in a design fit it into the ATF1500A. This strategy can be enabled for either all signals or for individual signals. These signals represent nodes in the design. This strategy will automatically convert sharable expander logic equations to foldback logic nodes. Sharable expander logic equations are defined with an implicit inversion while

foldback logic nodes for the ATF1500 require that this inversion be included in the logic equations. For example,

Sharable Expander Equation	**Foldback Logic Equation**
out = a & b; "Implicit Inversion	out = !(a & b); "Inversion must be defined

fit1500 design -strategy expander = anode,bnode

Converts sharable expander nodes *anode* and *bnode* to foldback logic nodes. All other nodes default to foldback logic nodes. An example of how to use this strategy is shown in Section 7.4 of this manual. This strategy defaults to being set *off*.

4.2.6 Node Collapsing [-strategy soft_buffer OFF| On = node_name,...,node_name]

This strategy allows the user to prevent nodes from being collapsed by the fitter. The fitter defaults to allowing collapsing of all nodes. This strategy can be enabled for either all signals or for individual signals. These signals represent nodes in the design. Preventing certain nodes from being collapsed can help a design to fit into the ATF15xx as well as the ATF1500A. Please refer to the section 7.6 of this manual for more information about how to use the node collapsing feature in your design. This option defaults to being set *off*.

fit1500 design -strategy soft_buffer = anode,bnode

Prevents the fitter from collapsing nodes *anode* and *bnode* respectively. All other nodes are allow to be collapsed by the fitter.

fit1500 design -strategy soft_buffer on

Prevents all nodes defined in the design from being collapsed.

4.2.7 Dedicated Inputs [-strategy dedicated_input ON | Off = pin_name,...,pin_name]

Use this strategy if you want the fitter to place certain input signals on the dedicated input pins of the ATF1500. This strategy can be enabled for up to 4 input signals or for individual signals. The actual number of input signals allowed will depend on the amount of global resources (pin clock, output enables, reset) on the ATF1500 that are used by the design. If dedicated inputs are already specified in the design file they will override whatever signals are defined by this strategy. This strategy defaults to being set *on*.

fit1500 design -strategy dedicated_input = a, b

Places the inputs *a* and *b* on any of the four dedicated input pins if they are available. Remaining dedicated input pins, if available, are assigned to design input signals depending on the number of times these inputs are used in the design.

fit1500 design -strategy dedicated_input off

Disables the fitter from placing any input signals on the dedicated input pins of the ATF1500.

4.2.8 Slew Rate Control [-str output_fast On | OFF = pin_name,...,pin_name]

This option allows the user to define the slew rate of the outputs of the ATF1500. The slew rate can be defined as either fast or slow. This option can be enabled for either all output signals or for individual signals. These signals represent output pins on the device. The fitter default to setting the slew rate for all outputs to be slow. This strategy defaults to being set *off*.

fit1500 design -strategy output_fast = a, b

Defines output signals *a* and *b* as being fast slew rate outputs. All other output signals are defined as slow slew rate outputs.

fit1500 design -strategy output_fast on

Defines all output pins to have fast slew rates.

4.2.9 Power Down Control [-str pd1=on] [-str sleep]

This option allows the power down pin to be used on the ATF15xx. This pin powers down the device to a zero power mode. The ATF15x series of devices has 2 such pins. Users can use either pd1 or pd2 pins to enable this mode. The ATF1500A fitter however uses the slep option. When this feature is enabled for the ATF1500A, pin 4(PLCC), pin 42 (TQFP) the associated macrocell is available for any buried logic functions such as foldback or cascade logic. This strategy defaults to being set *off*.

fit15xx design -str pd1 =on {Enables the power down pin on the ATF15xx}

fit15xx design -str pd2 =on {Enables the power down pin on the ATF15xx}

fit1500 design -strategy sleep {Enables the power down pin on the ATF1500A}

4.2.10 JEDEC File [-str jedec_file = file_name]

The option allows the user to instruct the fitter to save the JEDEC file as a different file name or to a different directory than the default directory. The fitter defaults by creating a [.jed] file and saving it in the working directory.

fit1500 design -strategy jedec_file = c:\fit1500\project\project.jex

The fitter will save the jedec file as *project.jex* in the *c:\fit1500\project* directory.

4.2.11 Vector File [-str vector_file = file_name]

This option allow the user to append different test vectors than the default test vectors, if included in the design file, to the JEDEC file created by the fitter.The vector file can be read as a different filename than the default filename or a different directory than the default directory. The fitter defaults to reading a [.tmv] file from the working directory.

fit1500 design -strategy vector_file = c:\fit1500\project\project.vec

Appends the vectors in the *project.vec* file in the *c:\fit1500\project* directory to the output JEDEC file *design.jed.*

4.2.12 Open Collector Outputs [str open_collector OFF|On = signal_name1,..]

To configure ALL outputs as Open Collector outputs, you can use the following property statement in your CUPL design:

PROPERTY ATMEL {open_collector = on};

To configure certain outputs as Open Collector outputs, you can use the following property statement in your CUPL design:

PROPERTY ATMEL {open_collector = [signal_name1],[signal_name2],....};

where [signal_name1&2] represents the actual signal names of the signals that you want to configure as output collector outputs.

4.3 Property Statement

The PROPERTY statement allow the user to specify any of the fitter options listed in sections (4.1-4.2) of this manual within the ABEL or CUPL design file instead of on the DOS command line. Examples of how to use the PROPERTY statement for ABEL and CUPL are included below.

4.3.1 ABEL

The syntax of the Property statement for ABEL is as follows

ATMEL Property 'fitter option';

The following ABEL source file illustrates the use of this statement. Note that a separate PROPERTY statement is required for each strategy. The keyword "str" is not required in the expression.

```
module fitter1;
title 'ATF1500A Fitter Example -- ATMEL PLD Applications';
fitter1 device 'p1500a'; "Device mnemonic is for ATF1500A in PLCC44

"************************************************************
"* ATF1500A ABEL Test Case Combinatorial Output     *
"* Buried Register Node, Pin clock, Pin reset           *
"************************************************************

"Property Statement must be included in the declarations section
"Each fitter strategy requires a separate property statement

ATMEL property 'keep';                "Keep pin pre-assignments
ATMEL property 'cascade_logic = mc1';   "Enable Cascade Logic feature for signal mc1
ATMEL property 'sleep ';                "Enable the power down pin to be used. The sleep option
                                        "is different from other ATF15xx (use PD1, PD2)

"Inputs

clk   pin 1;                                "Global Clock Signal
reset pin 2;                               "Global Reset Signal
mc2,mc3,mc4,mc5,mc6,mc7  pin 5,6,7,8,9,11;

"Outputs

mc1 pin 4 istype 'com';  "Define Buried Registered node for pin 4 macrocell

mc1fdbk node 77 istype 'reg_d,buffer';

Equations

mc1 = mc3 # mc4 # mc5 # mc6 # mc7 # mc1fdbk;

mc1fdbk.d = mc2 & mc5;
mc1fdbk.ck = clk;
mc1fdbk.ar = reset;

end fitter1
```

Figure 4-1 ABEL Property Statement Example

4.3.2 CUPL

The syntax for the property statement in CUPL which is used in the ATMEL-WinCUPL and PROCHIP tool is as follows,

PROPERTY ATMEL {fitter option };

The following CUPL example illustrates the use of this statement. Note that a separate PROPERTY statement is required for each strategy. The keyword "str" is not required in the expression.

```
Name        fitter1;
Date        10/01/01;
Company     Atmel;
Device      f1502isptqfp44;    /* TQFP Device Type for ATF1502AS*/

/**************************************************************/
/* ATF1502AS CUPL Test Case Combinatorial Output */
/* Buried Register Node, Pin clock, Pin reset         */
/**************************************************************/

/* All options need to be specified by a separate PROPERTY statements */

PROPERTY ATMEL { preassign=keep };                    /* Keep pin pre-assignments */
PROPERTY ATMEL { cascade_logic = mc1 }; /* Enable Cascade Logic for output signal mc1 */
PROPERTY ATMEL {pd1=on};                    /* Enable the Power Down pin to be used */
PROPERTY ATMEL {Logic_doubling=on}; /*Enable Logic Doubling */


/* Inputs */

Pin 37 = clk;   /* Global Clock Signal */
Pin 39 = !reset; /* Global Reset Signal */

/* Outputs */

pin 1 = mc1;    /* TQFP Pin assignments */
pin 2 = mc2;
pin 3 = mc3;
pin 5 = mc4;
pin 6 = mc5;
pin 7 = mc6;
pin 8 = mc7;

pinnode 77 = mc1fdbk;  /* Buried Register Node */

/* Logic Equations */

mc1 = mc3 # mc4 # mc5 # mc6 # mc7 # mc1fdbk;

mc1fdbk.d = mc2;
mc1fdbk.ck = clk;
mc1fdbk.ar = reset;
```

Figure 4-2 CUPL PROPERTY Statement Example

4.4 Fitter Macro Library for ATF1500A

An easy way to customize fitter operation in the design file is to use ATF1500 macro library. The fitter macro library includes macros which contain PROPERTY statements that specify fitter options. For example, to configure the fitter to accept sharable expander logic syntax you could include the EXP macro in your CUPL or ABEL source file. The ATF1500 macro library for both ABEL and CUPL is included in the fitter installation diskette. A listing of the macros in this library is shown in Table 2 below. Detailed descriptions of these macros along with their specific syntax is included in the Section 9.0 of this manual.

EXP Converts a Expander Node(s) to Foldback logic	**OPTIMIZE**- Turns on/off Logic Optimization Option
FOLDBACK_LOGIC -Defines a foldback Logic node(s)	**VECTOR_FILE**- Allows user to specify Vector file name
CASCADE_LOGIC- Enables Cascade Logic feature for specific output(s)	**CASCADE_LOGIC_ALL**-Enables/disables Cascade Logic feature for all outputs in the design.
XOR_SYNTHESIS- Turn on/off XOR Synthesis Option for specific output(s)	**XOR_SYNTHESIS_ALL**- Turn on XOR Synthesis Option for all outputs
GLOBAL -Assign specified input to dedicated input pins	**GLOBAL_ALL**- Enables/Disables Global input pin fitting assignments
TURBO- Sets all outputs to fast output slew rate	**TURBO_OUTPUT**- Sets a specific output to fast slew rate
SOFT- Prevent a specified node(s) from being collapsed	**SOFT_BUFFER_INSERTION** - Enable/disable Node Collapsing feature for all nodes in design
JEDEC_FILE- Allows user to specify JEDEC file name	**SLEEP**- Enables Pin Controlled Power Down mode for device

Table 4-2 ATF1500A Macro Library Listing

4.4.1 ATF1500A Macro Library for ABEL

To use the ATF1500A Macro Library in ABEL insert the following statement in your source file.

Library 'FIT1500';

The LIBRARY statement must be specified ahead of the individual macros you want to use. A portion of an ABEL source file is shown below to illustrate this.

```
module reset;   " Module Declaration"                        .
Title 'Example showing reset usage';

"Pin Declarations

data, clk  pin 2,3;
.

Library 'FIT1500';  "This includes the FIT1500.mac file which contains the ABEL ATF1500 Macros
                    "The Library statement is specified ahead of the specific macros you want to use
GLOBAL(reset);    "Macro- Use the dedicated global reset input for the reset signal
SLEEP(ON);        "Macro- Allow the power down pin to be used
.
.
Equations
```

Figure 4-3 ATF1500A Macro Library--ABEL Example

4.4.2 Using the CUPL Macro Library

To use the ATF1500A Macro Library for CUPL you need to copy the fit1500.m file from your <cupl_directory>\f1500 to your working directory. Then, insert the following statement in your CUPL source file.

$INCLUDE FIT1500.M

This statement must by specified ahead of the individual macros you want to use. Please refer to the CUPL file below for an illustration of this.

```
Name              reset;
Company           Atmel;
Device            f1500a;

/* Pin Declarations Sections */

pin 2 = data;
pin 3 = clock;
.
.
/* The Atmel Macro Library File for CUPL is included in the source file */
/* The fit1500.m file must reside in your working directory */

$INCLUDE FIT1500.M;

GLOBAL(reset];        /*This macro places the reset signal to the dedicated global reset pin */
SLEEP(ON);            /* This macro enables the power down pin to be used  */
.
```
.
/* Equations Section */

Figure 4-4 ATF1500A Macro Library--CUPL Example

The Fitting Process Flow

The fitter is a complex program which implements several options to make a design fit into the ATF1500A or the other members of the ATF15xx family. These options can be controlled by the user or selected automatically by the fitter. Please refer to section (5.1-5.3) for details on the number of design pass iterations which are perfomed before the fitter ultimately determines whether the design will fit. The fitting process is outlined in Figure 5-1 below
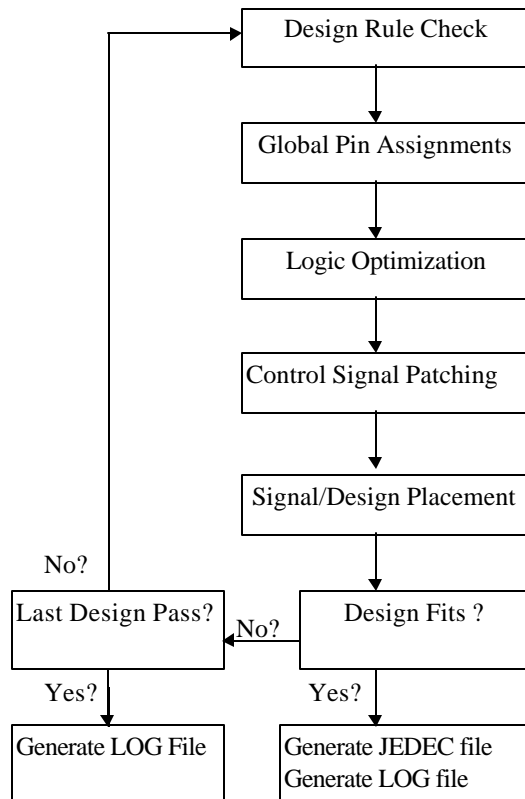
Figure 5-1. Fitting Process

_____

5.1 Design Passes

Each attempt the fitter makes to fit a design is called a design pass. Complex designs might require several design passes to fit. The fitter will only attempt a new design pass if the previous pass was not successful. The number of design passes the fitter uses depends on the preassignment option specified by the user.

5.1.1 Keep Option

When the Keep preassignment option is selected, the fitter will perform the design passes illustrated in Figure 5-2 below. Any pins or nodes not preassigned in the design will be automatically assigned by the fitter.

```
┌─────────────────────────────────────────────────┐
│ Pass 1 - Fit with user-specified preassignments and │
│           strategies only.                        │
└─────────────────────────────────────────────────┘
                              │
                              ▼
┌──────────────────┐   Yes?  ┌──────────────┐
│ Generate JEDEC file │◄──────│ Design Fits? │
│ Generate LOG file   │        └──────────────┘
└──────────────────┘                │
                              No?    ▼
┌─────────────────────────────────────────────────┐
│ Pass 2-  Ignore user-specified strategies         │
│           Keep original pin assignments            │
│           Re-fit Design                            │
└─────────────────────────────────────────────────┘
                              │
                              ▼
┌──────────────────┐  Yes? ┌──────────────┐  No? ┌──────────────────┐
│ Generate JEDEC file │◄────│ Design Fits? │─────►│ Generate LOG file │
│ Generate LOG file   │      └──────────────┘      └──────────────────┘
└──────────────────┘
```
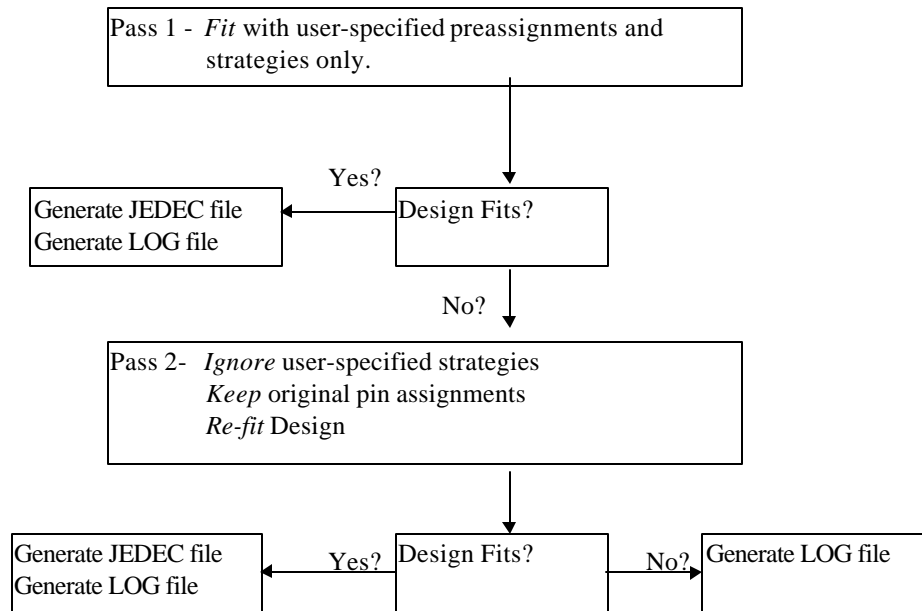
Figure 5-2. KEEP-Option-- Fitting Process

5.1.2 TRY Option

The TRY preassignment option is the default option selected by the fitter when no preassignment options are specified by the user. The fitter performs the design passes illustrated in Figure 5-3 below. The first two passes are identical to the KEEP preassignment option. The last pass is identical to the IGNORE preassignment option. Any pins that were not assigned in the design will be automatically assigned by the fitter.

```
┌─────────────────────────────────────────────────┐
│ Pass 1- Fit design with user-specified pin       │
│         assignments and strategies only          │
└─────────────────────────────────────────────────┘
                        │
                        ▼
┌──────────────────┐  Yes?  ┌──────────────────┐
│ Generate JEDEC file │◄─────│ Design Fits?     │
│ Generate LOG file   │      │                  │
└──────────────────┘        └──────────────────┘
                                    │ No?
                                    ▼
┌─────────────────────────────────────────────────┐
│ Pass 2- Ignore user-specified strategies where   │
│         necessary                                 │
│         Keep original pin assignments             │
│         Re-fit Design                             │
└─────────────────────────────────────────────────┘
                        │
                        ▼
┌──────────────────┐  Yes?  ┌──────────────────┐
│ Generate JEDEC file │◄─────│ Design Fits?     │
│ Generate LOG file   │      │                  │
└──────────────────┘        └──────────────────┘
                                    │ No?
                                    ▼
┌─────────────────────────────────────────────────┐
│ Pass 3- Ignore user specified pin assignments     │
│         and strategies                            │
│         Re-assign pins                            │
│         Re-fit Design                             │
└─────────────────────────────────────────────────┘
                        │
                        ▼
┌──────────────────┐ Yes? ┌────────────┐ No? ┌──────────────────┐
│ Generate JEDEC file │◄──│ Design Fits? │──►│ Generate LOG file │
│ Generate LOG file   │   │             │    │                  │
└──────────────────┘      └────────────┘     └──────────────────┘
```

Figure 5-3. TRY Option--Fitting Process

5.1.3 IGNORE Option

If the IGNORE option is selected the fitter will perform the design passes illustrated in Figure 5-4 below

```
┌─────────────────────────────────────────────────┐
│ Pass 1- Keep user specified strategies           │
│         If no strategies goto Pass 2, Otherwise  │
│         Re-assign pins                           │
│         Re-fit design                            │
└─────────────────────────────────────────────────┘
                          │
                          ▼
┌───────────────────┐  Yes?  ┌──────────────┐
│ Generate JEDEC file│◄───────│ Design Fits? │
│ Generate LOG file  │        └──────────────┘
└───────────────────┘              │
                                  No?
                                   ▼
┌─────────────────────────────────────────────────┐
│ Pass 2- Ignore user-specified strategies and pin │
│         assignments.                             │
│         Re-assign pins                           │
│         Re-fit design                            │
└─────────────────────────────────────────────────┘
                          │
                          ▼
┌───────────────────┐ Yes? ┌──────────────┐ No? ┌───────────────────┐
│ Generate JEDEC file│◄─────│ Design Fits? │────►│ Generate LOG file │
│ Generate LOG file  │      └──────────────┘     └───────────────────┘
└───────────────────┘
```
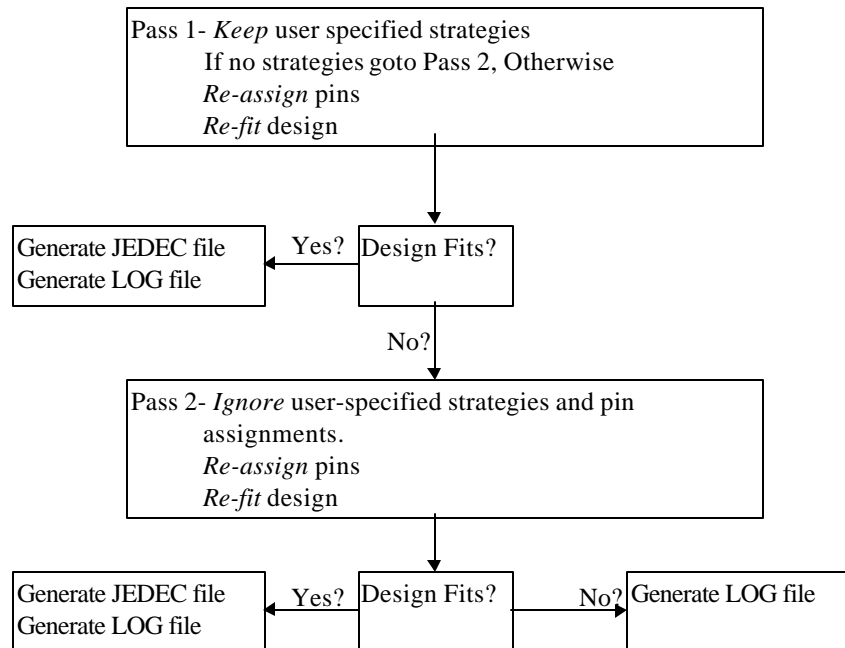
Figure 5-4. IGNORE Option--Fitting Process

_____

5.2 Design Rule Check

The first step in the fitting process is a Design Rule Check. In this stage the fitter checks for illegal pin assignments such as assigning an output to a dedicated input pin, whether there are too many outputs or inputs in the design, total number of product terms and registers being used, total number of buried resources needed etc. If the design uses more resources than are available in the ATF1500A or any of the ATF15xx devices, the fitter will fail Design Rule Check and indicate an error. In addition to checking the device resources the Design Rule Check will also remove all unused signals in the design.

_____
_

5.3 Assign Global Input Pins

In this fitting step the fitter evaluates whether any global resources are used in the design. Global input pins, such as the clock, reset or output enables, if defined in the design, are assigned first. Any Global inputs pins left over are then assigned to any user-specified input signal defined by the dedicated input strategy described in section 4.2.7 of this manual. If global input pins still remain unassigned or if the dedicated input strategy is not specifed, input signals are assigned to these pins based on how frequently they are used in the design.

_____
_

5.4 Logic Optimization

After Global input pins have been assigned the fitter performs logic optimization to efficiently map design logic into the ATF15xx device. The process flow is shown below.
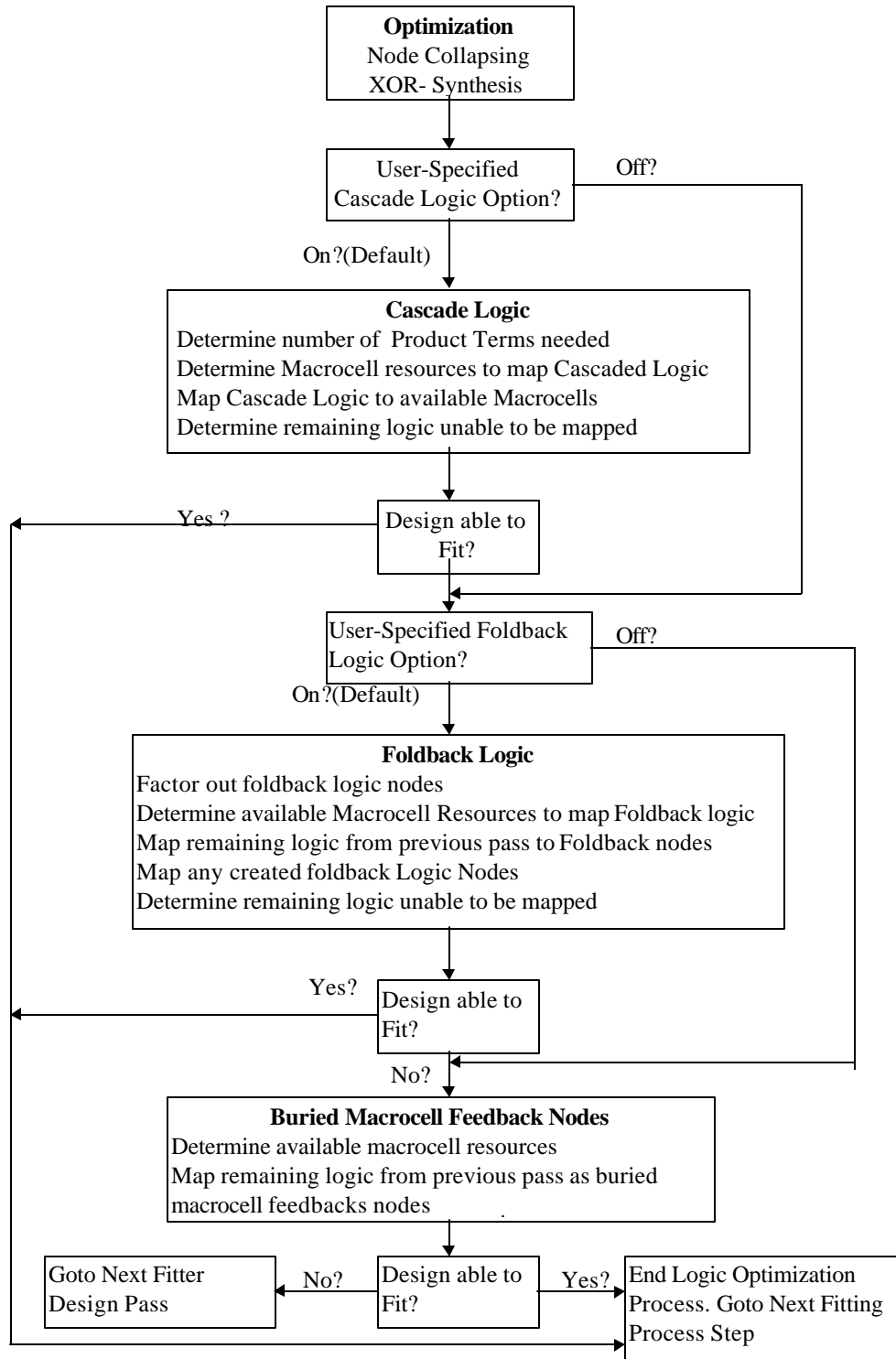
Figure 5-5. Logic Optimization Process

_____
—

5.5 Control Signal Patching

The Control Signal Patching process is performed when a design has multiple product term flip-flop clocks, clock enables, resets, presets or output enable logic. The fitter will create foldback or buried feedback nodes to map these inputs into the ATF15xx. This process is necessary since the ATF15xx macrocell has only one product term available for each of these inputs.

_____
—

5.6 Signal/Design Placement

During the Signal and Design Placement step the fitter places both preassigned and unassigned signals into the ATF1500A and then maps the design logic with these signal placements into the device. Preassigned signals are placed on the pins and nodes defined by the user. Unassigned signals are placed automatically by the fitter. Any nodes created by the fitter during the fitting process are also placed during this step. After all signals are placed, the fitter evaluates the resources required to fit the design logic into the device and maps the design logic into the architecture of the ATF15xx. During the mapping process the fitter determines whether the logic is using invalid macrocell configurations, is unable to fit into the macrocells of the ATF15xx, is using the wrong register types or register control functions. (i.e. Synchronous Preset) or is using too many resources in the device. If the design logic does not map into the ATF15xx this step is repeated in a new design pass. If this process fails on the last design pass the fitter will terminate the fitting process and indicate that the design does not fit. If the Signal/Design placement process is successful for any design pass the fusemapping process described in section 5.7 will be performed.

_____
———

5.7 Fusemapping

The is the last step in the fitting process if the Design/Signal Placement process is successful. A final signal and design placement check is performed to ensure correct mapping and then the fitter generates a JEDEC file for the ATF15xx device.

_____
══

Interpreting the Fitter Log File

The ATF15xx fitter creates a Log file (*.fit) that reports on the fitting process. This log file is created whether the design fits or not. If the design does not fit the fitter will indicate errors in the logic file. If the design fits a JEDEC (.JED) file will also be created along with the Log file. The Fitter log file contains the following sections.

- Initial Fitting Strategy and Properties
- Global Pin Assignments
- Input/Output Pin Preassignments
- Control Signal Patching
- Floating (unassigned) Signal Placement
- Control Signal Assignment
- Programmed Logic
- Pin Layout and Listing
- Logic Resources Usage

Figure 6-1 below shows a sample Log file from the CUPL source file listed in Figures 6-2 and 6-3

```
Atmel ATF1500 Fitter Version 2.41  , running Mon Nov 20 16:12:25 1997

fit1500 logic.tt2 -dev p1500 -log logic.fit -pre try -o logic.tt3 -errlog adhoc.err
```

This line specifies the fitter option selected by the user

```
Project fits successfully
```

Summary of the fitting process. Design fits successfully.

```
****** Initial fitting strategy and property ******
pla_in_file = logic.tt2
pla_out_file = logic.tt3
jedec_file = logic.jed
vector_file = logic.tmv
log_file = logic.fit
err_file = adhoc.err
device_name = p1500
package_type = PLCC
preassign_file =
property_file =
preassignment = TRY
silent = FALSE
sleep_mode = FALSE
security_mode = FALSE
swap_clk_ce = TRUE
p1500dip48 = FALSE
dedicated_input_clock = UNASSIGNED
dedicated_input_reset = UNASSIGNED
dedicated_input_oe = UNASSIGNED
supporter = ABEL
optimize = SPEED
soft_buffer = OFF
xor_synthesis = ON
foldback_logic = ON
```

This section describes the initial fitting properties and strategies that will be used by the fitter on the first attempt to fit the design. These options are either set by the user or set to the fitter's default conditions. Note that the options Swap_clk_ce and p1500dip48 are Atmel Internal options not available to users.

```
expander:  x0  x1  x2
cascade_logic = ON
dedicated_input = ON
output_fast = OFF
***************************************************


-------------------------------------------------------------------
Fitter_Pass 1,  Preassign = KEEP ,  Strategy = KEEP ,  Algorithm = 1
-------------------------------------------------------------------


Performing global pin assignments ...
---------------------------------------
out3 :
            output enable(positive polarity) is forced to product term
            clock is forced to product term
            reset is forced to product term

Final global control pins assignment (if applicable)...
-------------------------------------------------------
clock assigned to pin 43
reset assigned to pin 1
I5 assigned to pin 2

Performing input pin pre-assignments ...
---------------------------------------
I1 is placed at pin 44
I5 is placed at pin 2
clock is placed at pin 43
reset is placed at pin 1

out3.C equation needs patching.
out3.AR equation needs patching.
2 control equations need patching

Performing output pin pre-assignments ...
-----------------------------------------
regnode is placed at node 78 (MC 55)
out2 is placed at pin 5 (MC 2)
out3 is placed at pin 6 (MC 3)
out4 is placed at pin 7 (MC 4)

Attempt to place floating signals ....
--------------------------------------
out5 is placed at pin 11 (MC 7)
out1 is placed at pin 24 (MC 32)
out3_1 is placed at feedback node 107 (MC 31)
X2 is placed at feedback node 106 (MC 30)
X0 is placed at foldback expander node 76 (MC 32)
out3_0 is placed at feedback node 105 (MC 29)
X1 is placed at feedback node 104 (MC 28)
I4 is placed at pin 28 (MC 28)
I3 is placed at pin 27 (MC 29)
I2 is placed at pin 26 (MC 30)
I0 is placed at pin 25 (MC 31)
Creating JEDEC file logic.jed ....

Final control functions assignment (if applicable)...
-----------------------------------------------------
out3.C uses product term
out3.AR uses product term
out3.OE uses product term
```

The fitter will keep preassignments on the first design pass. Note that the description of all the fitter passes are included in the fitter log file. If the design fits, then the last fitter pass is applicable. If the design failed to fit, then the fitter pass descriptions can provide insights on why a

The fitter performs an initial placement for all global inputs.. For output *out3* the output enable is forced to a product term because the OE is active high. (The pin controlled Output Enable must be active low). Clock and reset are initially placed to product terms, until a final placement of all global

Final placement of all global inputs completed. Note that the reset and clock signals are assigned to the global clock and reset pins. I5 is preassigned in the design file to pin 2.

Pre-assigned inputs are placed. A summary for all pre-assigned inputs is shown. Note that there was one Global input pin , pin 44, was leftover so it is assigned to *I1*. This signal is used frequently in the design.

Control Signal Patching is performed here. The Reset and Clock inputs for *out3* are more than one product term. Therefore, the fitter will create nodes for these

All Pre-assigned output pins and nodes are placed. For example *out2* is placed on pin 5 (Macrocell 2).

All inputs,outputs and nodes not assigned in the source file are placed., this also includes nodes created by the fitter during the control patching

Note that X0 has been placed on an foldback node. X2, although originally defined as an foldback node is placed as a buried feedback node since it requires two product terms.

The signal and design placement step is successful so the fitter is creating a JEDEC file with the name *logic.jed* after a final placement check is performed.

The final placement of all flip-flop control or output enable signals is performed here. The fitter determines whether global input pins or I/O are used

Programmed logic section. This logic represents the actual logic programmed into the device. This logic may not be in the same form as the user-specified logic since the fitter optimizes the logic to make the design fit efficiently into the

```
out4.C uses global GCLK pin
out4.AR uses global GCLR pin
regnode.C uses global GCLK pin
regnode.AR uses global GCLR pin
```

Fitter converted the X0 expander logic equation to ATF1500 foldback

```
p1500 programmed logic:
--------------------------------------------------

out1  = !I1 & !I3
           # X0  ;
!X0  = !I1 & I0  ;
```

These are the nodes created by the fitter during the Control Signal Patching section to patch the Clock and Reset signals for signal *out3*.

```
out2  = X1  ;
!X1  = !I0 & !I3 & I2  ;
```

XOR Synthesis function. The logic equation for *out4* is mapped to the hardware XOR in the macrocell. _EQ1 and _EQ2 represent the XOR inputs.

```
out3.D  = I1 & I0  ;
out3.C  = out3_0  ;
out3_0  = I2
           # I3 & I4  ;
out3.AR = out3_1  ;
out3_1  = X0
           # X1  ;
out3.OE = I5  ;
```

This logic implements the Cascade Logic feature. Product terms are borrowed from an adjacent macrocell .Please refer to the usage table for more information.

```
out4.D  = (_EQ0 $ _EQ1)
_EQ0 = X1  ;
_EQ1 = X2  ;
!X2  = X0
           # X1  ;
out4.C = clock  ;
out4.AR = !reset  ;

out5  = !I1 & I0 & I3 & !I4 & I5
           # I1 & !I0 & I3 & !I4 & I5
           # I3 & I2 & !I4 & I5
           # I1 & I0 & !I3 & !I2 & !I4 & I5
           # !I1 & !I0 & !I3 & !I2 & !I4 & I5
           # !I1 & I0 & I3 & !I5
           # I1 & !I0 & I3 & !I5
           # !I3 & I2 & !I5
           # I1 & I0 & I3 & !I2 & !I5
           # !I1 & !I0 & I3 & !I2 & !I5
           # I4 & !I5  ;
```

Node collapsing feature. The fitter collapsed nodes XOR1 and XOR2 to efficiently fit them into ATF1500's macrocell. These nodes must be defined so the fitter could perform XOR synthesis. Refer to section 7.5 for more information.
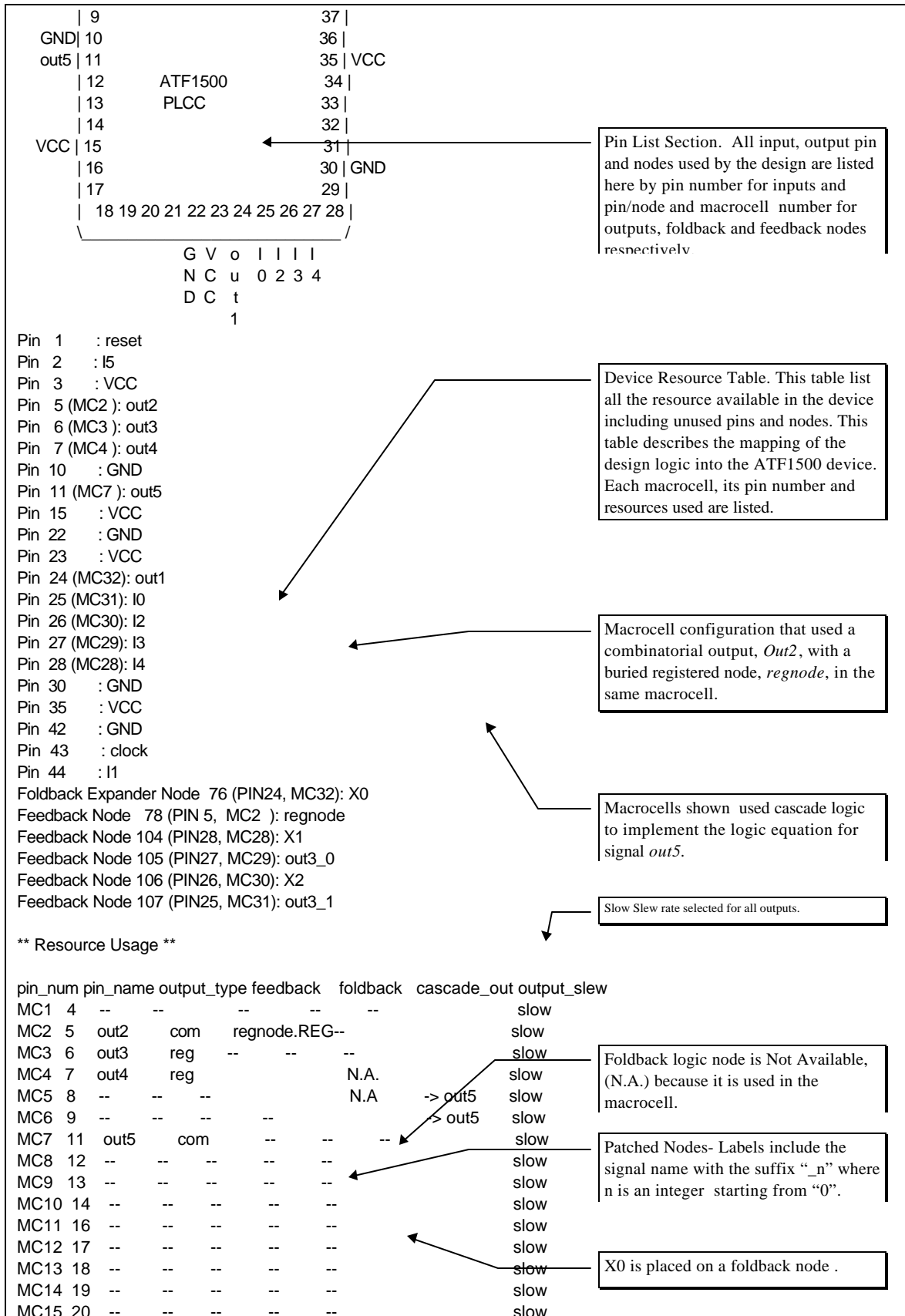
```
regnode.REG  = X2  ;
regnode.C = clock  ;
regnode.AR = !reset  ;

*** xor1 is collapsed (XOR) ***
*** xor2 is collapsed (XOR) ***
```

Pin Layout Section. The locations of all assigned signals on the device are specified.

```
p1500  PLCC Pin Layout:
                      r    c
           o  o          e     l
           u  u    V    s    o  G
           t  t    C l  e l  c  N
           3  2    C 5  t 1  k  D
       _____
       /  6  5  4  3  2  1 44 43 42 41 40    \
   out4 |  7                             39 |
        |  8                             38 |
```

```
                | 9                     37 |
         GND| 10                     36 |
         out5 | 11                     35 | VCC
             | 12        ATF1500       34 |
             | 13        PLCC          33 |
             | 14                     32 |
         VCC | 15                     31 |
             | 16                     30 | GND
             | 17                     29 |
             | 18 19 20 21 22 23 24 25 26 27 28 |
              _____/

                      G V  o  l l l l
                      N C  u  0 2 3 4
                      D C  t
                           1
```

Pin List Section. All input, output pin and nodes used by the design are listed here by pin number for inputs and pin/node and macrocell number for outputs, foldback and feedback nodes respectively.

```
Pin  1     : reset
Pin  2     : I5
Pin  3     : VCC
Pin  5 (MC2 ): out2
Pin  6 (MC3 ): out3
Pin  7 (MC4 ): out4
Pin 10     : GND
Pin 11 (MC7 ): out5
Pin 15     : VCC
Pin 22     : GND
Pin 23     : VCC
Pin 24 (MC32): out1
Pin 25 (MC31): I0
Pin 26 (MC30): I2
Pin 27 (MC29): I3
Pin 28 (MC28): I4
Pin 30     : GND
Pin 35     : VCC
Pin 42     : GND
Pin 43     : clock
Pin 44     : I1
Foldback Expander Node  76 (PIN24, MC32): X0
Feedback Node  78 (PIN 5,  MC2 ): regnode
Feedback Node 104 (PIN28, MC28): X1
Feedback Node 105 (PIN27, MC29): out3_0
Feedback Node 106 (PIN26, MC30): X2
Feedback Node 107 (PIN25, MC31): out3_1
```

Device Resource Table. This table list all the resource available in the device including unused pins and nodes. This table describes the mapping of the design logic into the ATF1500 device. Each macrocell, its pin number and resources used are listed.

Macrocell configuration that used a combinatorial output, *Out2*, with a buried registered node, *regnode*, in the same macrocell.

Macrocells shown used cascade logic to implement the logic equation for signal *out5*.

Slow Slew rate selected for all outputs.

** Resource Usage **

pin_num	pin_name	output_type	feedback	foldback	cascade_out	output_slew
MC1 4	--	--	--	--	--	slow
MC2 5	out2	com	regnode.REG	--		slow
MC3 6	out3	reg	--	--	--	slow
MC4 7	out4	reg		N.A.		slow
MC5 8	--	--	--	N.A	-> out5	slow
MC6 9	--	--	--		> out5	slow
MC7 11	out5	com	--	--	--	slow
MC8 12	--	--	--	--	--	slow
MC9 13	--	--	--	--	--	slow
MC10 14	--	--	--	--	--	slow
MC11 16	--	--	--	--	--	slow
MC12 17	--	--	--	--	--	slow
MC13 18	--	--	--	--	--	slow
MC14 19	--	--	--	--	--	slow
MC15 20	--	--	--	--	--	slow

Foldback logic node is Not Available, (N.A.) because it is used in the macrocell.

Patched Nodes- Labels include the signal name with the suffix "_n" where n is an integer starting from "0".

X0 is placed on a foldback node .

```
MC16  21   --      --     --    --       --                slow
MC17  41   --      --     --    --       --                slow
MC18  40   --      --     --    --       --                slow
MC19  39   --      --     --    --       --                slow
MC20  38   --      --     --    --       --                slow
MC21  37   --      --     --    --       --                slow
MC22  36   --      --     --    --       --                slow
MC23  34   --      --     --    --       --                slow
MC24  33   --      --     --    --       --                slow
MC25  32   --      --     --    --       --                slow
MC26  31   --      --     --    --       --                slow
MC27  29   --      --     --    --       --                slow
MC28  28   I4      --     X1          N.A.    --           slow
MC29  27   I3      --     out3_0 --       --               slow
MC30  26   I2      --     X2               --    --        slow
MC31  25   I0      --     out3_1 --       --               slow
MC32  24   out1    com                 X0      --          slow
MC0   43   clock   --     --            N.A.    --          slow
MC0   1    reset   --     --            N.A.    --          slow
MC0   44   I1      --     --            N.A.    --          slow
MC0   2    I5      --     --            N.A.    --          slow


Logic Array Block  Logic Cells  I/O Pins    Foldbacks   Cascades
A: LC1  - LC16     4/16( 25%)   4/16( 25%)  0/16( 0%)   2
B: LC17 - LC32     5/16( 31%)   5/16( 31%)  1/16( 6%)   0


Total dedicated input pins used:     4/ 4 (100%)
Total I/O pins used          9/32 (28%)
Total logic cells used       9/32 (28%)
Total Flip-Flop used         3/32 (9%)
Total foldback logic used    1/32 (3%)
Total cascade used           2
Total input pins             8
Total output pins            5
Creating pla file logic.tt3 with 15 inputs 20 outputs, 13 pins 8 nodes and 35 terms...
---------------  End fitter, Design FITS
$DEVICE p1500 fit
```

Resources Usage Summary. This section summarizes the total macrocell and logic resources used by the entire design.

Figure 6-1. Sample Fitter Log File

Fitter Hints

This section discusses hints on how to use the fitter to fit your designs efficiently, to optimize your design for performance to reduce system noise and power consumption. The hints are listed below.

- Do not Assign Pins
- Use Default Properties and Strategies
- Use Cascade Logic for Design Performance
- Using Foldback Logic
- Use XOR Synthesis
- Preventing Nodes from Collapsing
- Using Slew Rate Control
- Using the Power Down Pin

7.1 Hint 1 - Do not assign pins

Although the fitter is very efficient in utilizing the resources in the ATF15xx, poorly chosen pin and node assignments can prevent a design from fitting. The fitter employs many pin assignment and logic optimization strategies to fit a design into the ATF15xx. Global input are usually assigned to control pins such as the clock, reset or output enables to reduce product term usage in the macrocells. I/O pins are allocated for logic only after all foldback logic nodes and cascade logic resources in the macrocells are fully utilized. As a result, the fitter will optimize as much logic into the fewest number of macrocells as possible and choose its pin assignments to reflect this goal. This process assures that the logic is allocated for the best fit into the device. Poorly chosen pin assignments impose additional restrictions on the fitting process thereby forcing the fitter to tailor the logic around the pin assignments.

7.2 Hint 2 - Use Default Properties and Strategies

The default properties and strategies the fitter uses have been selected to generate the best fit for the design into the ATF15xx. If you want to customize properties and strategies to meet your particular design needs you are free to do so, and the methods to do this have been discussed in Section 4.0 of this manual. The tradeoff is that whenever you customize fitter operation by using specific design strategies you are constraining the fitter to fit the design around these strategies and properties and this may result in a sub-optimal fit. For more information refer to Section 5.1 of this manual.

7.3 Hint 3 - Using Cascade Logic for Design Performance

If performance is an issue in your system design, it is a good idea to use the cascade logic to borrow product terms from adjacent macrocells. This is useful, for high performance state-machines. Using cascade logic incurs a much lower propagation delay than using foldback nodes or buried macrocell nodes. By default, the fitter will use cascade logic to optimize a design for performance first. If the design fails to fit, the fitter uses foldback logic nodes or buried macrocell feedback nodes to implement logic. If the propagation delay of certain outputs is important to meet your system timing requirements, you can customize the fitter to use cascade logic for specific outputs. Please refer to Figures 13 and 14 for ABEL and CUPL examples of this feature. It is important to choose pin assignments that place cascade logic on macrocells which can borrow product terms from adjacent macrocells. For example, each macrocell except corner cells (See Figure 1) can borrow up to 5 product terms from its adjacent macrocell. Additional macrocells can be cascaded to provide up to 40 product terms for the macrocell at the bottom of the cascade chain.

7.4 Hint 4 - Using Foldback Logic

The fitter uses foldback logic as an optimization strategy to fit a design if cascade logic fails. Preventing the fitter from using foldback logic nodes may prevent a design from fitting. Allowing the fitter to use these nodes can provide a more efficient fit for your design. The examples shown in Figure 7-1 and 7-2 show ABEL and CUPL examples respectively, on how to assign foldback logic nodes and convert expander equations to foldback logic.

```
module  logic
title 'Example showing various features of the ATF1500A';

logic device 'p1500a';

library 'fit1500';                          "Includes ABEL Macro Library into source file

EXP(X0);                                    "Macro- converts expander nodes to foldback logic nodes
CASCADE(out5);                              "Macro - specifies Cascade logic to be used for signal out5.
                                             "Node assignment for these expanders are shown below
I4,I3,I2,I1,I0  pin ;                       "Non Pre-assigned signals
I5          pin 2;                          "Input pin preassignment
clock       pin 43;                         "Global clock preassignment
!reset      pin 1;                          "Global reset preassignment

"outputs
out1        pin   istype 'com';                     "Non-preassigned signal
out2        pin 5 istype 'com';
out3        pin 6 istype 'reg';            "Pre-assigned outputs
out4        pin 7 istype 'reg';
out5        pin 11 istype 'com';

"Sharable Expander nodes
X0          node  istype 'com';                     "Expander node assignment

"Nodes for XOR Synthesis          "To use XOR Synthesis feature these nodes must be declared
xor1        node istype 'com';              "They are collapsed when the fitter maps the design.
xor2        node istype 'com';

"Buried Registered Node
regnode     node 78 istype 'reg';          "This is a buried registered node located in the same macrocell
                                             " as out2,  which is a combinatorial output.
"Buried Combinatorial Nodes
X1,X2       node  istype 'com';

equations

"Implement Sum of Products using expander terms by DeMorgan's inversion,
" X2 = X0 # X1.                    Expander Logic equations are defined for node X0
                                             "The EXP macro will convert the X0 equation to a foldback logic node.
X0 = (!I1 & I0);                            "Note  the expander logic equation is defined with an
X1 = (!I3 & I2 & !I0);                      "implicit inversion.
X2 = (X0 # X1);
out1  =   X0 # !I3 & !I1;

out2     = X1;
regnode.d = X2;
regnode.ck = clock;
regnode.ar = reset;

"Implement Registered Logic requiring Soft-Buffer Patching
```

```
out3.d  = I0 & I1;
out3.ck = I2 # (I3 & I4);                "Multiple Product Term Clock requires Patching
out3.ar = X0 # X1;                       "Multiple Product Term Reset requires Patching
out3.oe = I5;                            "Product- term-controlled Output Enable
"Implement XOR Synthesis feature
xor1 = X1;                               "Single Product Term XOR Synthesis equations
xor2 = X2;

out4.d = xor1 $ xor2;                    "Hardware XOR will be used to implement this logic
out4.ck = clock;
out4.ar = reset;

"Implement Cascade Logic Feature
"Borrow product terms from adjacent macrocell

out5 = (I0 $ I1) #  (I2 $ I3) #  (I4 $ I5);

"Functionally test a portion of the design and insert test vectors in JEDEC file

"When running PLASIM in ABEL make sure original expander equations are used. PLASIM assumes an implicit inversion
"of the logic. Only include the EXP macro to convert expander equations to foldback nodes when ready to fit the design
"and create a JEDEC file. Otherwise,  PLASIM will fail.

test_vectors
([I3, I2, I1, I0] -> [out1, out2])
 [0,0,1,1] -> [0,0];
 [1,0,0,0] -> [0,0];
 [1,1,0,0] -> [0,0];
 [0,0,0,0] -> [1,0];
 [0,1,1,1] -> [0,0];
 [0,1,0,1] -> [1,1];
 [0,1,0,0] -> [1,1];
 [0,0,1,0] -> [0,0];
 [1,0,0,1] -> [1,1];
 [0,1,1,0] -> [1,1];

 end
```

Figure 7-1. ABEL Source File Example

The CUPL example, Figure 7-2, follows.

```
Name              logic;
Device            f1500a;

$INCLUDE fit1500.m;     /* ATF1500A Fitter Macro Library file include within source file */

EXP(X0);                /* Macro-  Defines X0 as a sharable expander node and converts it to foldback logic */
CASCADE(out5);          /* Macro-  Specify cascade logic for signal  out5 */

/* Inputs */

pin  =  I0;       /* Signals I0 though I5 will be assigned pins by the fitter */
pin  =  I1;
pin  =  I2;
pin  =  I3;
pin  =  I4;
pin 2 = I5;
pin 43 = clock;
pin 1 = !reset;
```

```
/* Outputs */

pin = out1;                /*  Signal out1 will be assigned an I/O pin by the fitter */
pin 5 = out2;
pin 6 = out3;
pin 7 = out4;
pin 11 = out5;
/* Sharable Expander Node*/

pinnode  =  X0;               /*The fitter will determine where to place sharable expander node X0 */

/* XOR Synthesis Nodes */

pinnode = xor1;               /* Signals xor1 and xor2 must be defined as nodes for XOR Synthesis */
pinnode = xor2;

/* Buried Register Nodes */

pinnode 78 = regnode;         /* Signal regnode defined a buried registered node for the same  */
                              /* macrocell where out2 is defined as a combinatorial output */

/*Equations */

X0 = (!I1 & I0);
X1 = (!I3 & I2 & !I0);
X2 = (X0 # X1);

out1 = X0 # !I3 & !I1;
out2 = X1;
regnode.d = X2;
regnode.ck = clock;
regnode.ar = reset;

out3.d = I0 & I1;
out3.ck = I2 # (I3 & I4);
out3.ar = X0 # X1;
out3.oe = I5;

xor1 = X1;
xor2 = X2;

out4.d = xor1 $ xor2;
out4.ck = clock;
out4.ar = reset;

out5 = I0 $ I1 # (I2 $ I3) # (I4 $ I5);

/* Test vectors for CUPL must be specified in a simulation input [.so] file */
/* CUPL's functional simulator CSIM will simulate the design using this file and generate an */
/* output [.so] file. CUPL wil automatically append the simulation output in the [.so] file as test vectors to */
/* the JEDEC file created by the fitter. You must specify at least the -js option when compiling your */
/* source file in CUPL to generate test vectors in the JEDEC file. */
```

Figure 7-2. CUPL Source file Example

7.5 Hint 5- Use XOR Synthesis

When your design includes any type of XOR logic, the XOR synthesis feature can be helpful in reducing product terms. This feature uses the hardware XOR gate within the ATF1500A's macrocell to implement

XOR logic. The XOR feature is, by default, enabled by the fitter. If the XOR synthesis strategy is not used, the design logic is mapped by the fitter into a sum of products form which may generate many product terms. This may prevent the design from fitting into the ATF1500A. You can help the fitter by making sure this feature is enabled, examining the intermediate equations feeding XOR terms into your logic and defining these equations as nodes. Figures 7-3 and 7-4 respectively, show ABEL and CUPL examples of how to use XOR Synthesis.

```
module xorsyn
title 'ATF1500A Xor Synthesis Example';
xorsyn device 'p1500a';

"Inputs

A0,A1,A2,A3,A4,A5,A6,A7  pin ;

"Outputs
Y0                 pin istype 'com';
xornode1           node;
xornode2           node ;

"Assign each intermediate logic function feeding an XOR to a node. Let the fitter assign node numbers.
"The fitter may either collapse these nodes or retain them at its discretion. Let the fitter choose how to
"implement the logic.  It will determine the most efficient method to treat these nodes.

Equations

"Original Logic Implementation

"  Y0 =  ((A0 & A1) $ (A2 & A3)) $ ((A4 & A5) $ (A6 & A7));

"Note that the XOR for the A0 through A3 inputs is an intermediate logic
"function that could be assigned to a node. The same is true for the A4 though A7 inputs.

"The original logic implementation when reduced to sum of products produces 40 product terms
"and uses 8 macrocells on the ATF1500 or about 25% of all available resources .

"XOR Synthesis Implemention

xornode1 = (A0 & A1) $ (A2 & A3);   "Separate Node equations for first level XOR
xornode2 = (A4 & A5) $ (A6 & A7);

Y0 = xornode1 $ xornode2;         "Hardware XOR implemented for XOR node equations

"The above implementation uses two macrocells or only 6% of all available macrocell resources. One macrocell  was
used for "xornode1. The fitter collapsed the xornode2 node into the second macrocell and used the hardware XOR for
the output.

end xorsyn;
```

Figure 7-3. XOR Synthesis ABEL Example

The CUPL example for XOR Synthesis is shown in Figure 7-4.

```
Name xorsyn;
Device f1500a;

/* Inputs */

Pin = A0;
Pin = A1;
Pin = A2;
Pin = A3;
Pin = A4;
Pin = A5;
Pin = A6;
Pin = A7;

/* Outputs */

Pin = Y0;

/* XOR Synthesis Nodes */

pinnode  = xornode1;
pinnode = xornode2;

/* Assign each intermediate logic function feeding an XOR input to a node. This */
/* will prompt the fitter to use XOR synthesis.  Let the fitter assign node numbers. */
/* It will determine the most efficient method to treat these nodes. */

/* Logic Equations */

/* The original function to be implemented is */
/* Y0 = ((A0 & A1) $ (A2 & A3)) $ ((A4 & A5) $ (A6 & A7)); */

/* To simply the implemention of this function on the ATF1500  represent */

xornode1 = (A0 & A1) $ (A2 & A3);
xornode2 = (A4 & A5) $ (A6 & A7);

Y0 = xornode1 $ xornode2;

/* This implementation only uses two macrocells  verses 8 macrocells */
/* or 40 product terms with the original function */
```

Figure 7-4. XOR Synthesis CUPL Example

7.6 Hint 6- Prevent certain nodes from Collapsing

In certain designs it may be necessary to prevent the fitter from collapsing certain nodes. These nodes when collapsed may generate too many product terms and could prevent a design from fitting into the ATF1500A. You can help fitter by inserting nodes in specific places in your design and preventing the fitter from collapsing these nodes. The following are useful guidelines for where to insert these nodes in a design.

- Outputs of complex intermediate combinatorial logic expressions
- Intermediate logic expressions which feed many different inputs
- Multiple product-term register clock, reset or preset functions
- Muliple product-term output enables.

Figure 7-1, which list the *logic.abl* source file, shows where the fitter automatically inserted nodes for multiple product-term register clock and reset terms during the control patching section of the fitting process. The following examples in Figures 7-5 and 7-6 respectively show ABEL and CUPL examples of where to insert nodes in a design to help the fitting process.

```
module softbuf
title 'Example Illustrating of Node Collapsing';
softbuf device 'p1500at';

"This file implements a 4 bit synchronous up counter with data load. The counter will be loaded with data from the data
bus "when data from the a_data and b_data busses compare. Otherwise the counter is free-running.  The terminal
count is latched "by a flip-flop with a multiple product term preset. The counter outputs are controlled by a multiple
product term output enable. "The counter load function is the output of a complex combinatorial expression which feeds
each counter output bit.

"Inputs
clock         pin ;
!reset        pin ;
en1           pin ;
en2           pin ;
pre1          pin ;
pre2          pin ;
d0,d1,d2,d3   pin ;
a0,a1,a2,a3   pin ;
b0,b1,b2,b3   pin ;

"Outputs
cnt0,cnt1,cnt2,cnt3 pin istype 'reg_d,buffer';   "Complex Registered Output
tc                  pin istype 'reg_d,buffer';
out_enable          node ;                "Multiple Product term Output Enable
preset              node ;                "Multiple Product term Preset
load                node ;                "Complex Combinatorial Expression.

"The load expression is assigned a node since it is the output of a complex combinatorial expression, which feeds into
each of "the counter output logic. If this node were collapsed and included within the counter logic. The counter logic
equations would "contain many (approx. 20 on some outputs) product terms. The preset and out_enable nodes are
examples of multiple
"product term register preset and counter output enable product terms. These equations should be declared as nodes.

"Set Variable Definitions
data = [d0,d1,d2,d3];
count= [cnt3,cnt2,cnt1,cnt0];
a_data = [a0,a1,a2,a3];
b_data = [b0,b1,b2,b3];

Atmel Property 'soft_buffer = load';
"Prevents Load  soft buffer node from being collapsed. The preset and out_enable nodes are treated at the fitter's
discretion.

Equations

"Output Equations

count.d = (count.fb + 1) & !load   "Simplified Expression for counter output logic. Note that load is used in every output.
     # (data) & load;
count.ar = reset;
count.ck = clock;
count.oe = out_enable;

tc.d = cnt0 & cnt1 & cnt2 & cnt3;
```

```
tc.ck = clock;
tc.ar = reset;
tc.ap = preset;        .

"Intermediate expressions

"These expresions are defined as soft buffer nodes

load = (a_data == b_data);       "Simplified expression of 4 bit compator logic.
out_enable = en1 $ en2;      "Two product term enable equation selecting either en1 or en2
preset = pre1 $ pre2;        "Two product term flip-flop preset equation selecting either pre1 or pre2

end softbuf;
```

Figure 7-5. Node Collapsing - ABEL Example

The CUPL example for the same file follows,

```
Name softbuf;
Device f1500at;

 /* Inputs */

 Pin = clock;
 Pin = !reset;

 Pin = en1;  /* Output Enable inputs */
 Pin = en2;

 Pin = pre1;  /* Preset Inputs */
 Pin = pre2;

 Pin = d0;     /* Data bus Inputs */
 Pin = d1;
 Pin = d2;
 Pin = d3;

 Pin = a0;   /* A-bus inputs */
 Pin = a1;
 Pin = a2;
 Pin = a3;

 Pin = b0;   /* B-bus inputs */
 Pin = b1;
 Pin = b2;
 Pin = b3;

/* Outputs */

 Pin = cnt0;   /* Counter Outputs */
 Pin = cnt1;
 Pin = cnt2;
 Pin = cnt3;

 Pin = tc;          /*Terminal Count */

 /* Buried Nodes */

 pinnode = out_enable;
```

```
 pinnode = preset;
 pinnode = load;

 Property Atmel {soft_buffer = load};

/* Prevents signal 'load' from being collapsed. Other Nodes  are treated at the fitter's discretion.  */

/* Logic Equations */
/* Pre-loadable Up counter Logic Equations */

cnt0.D  = !cnt0 & !load # load & d3  ;
cnt0.CK = clock  ;
cnt0.AR = !reset  ;
cnt0.OE = out_enable  ;

cnt1.D  = !cnt1 & cnt0 & !load
          # cnt1 & !cnt0 & !load
          # load & d2  ;
cnt1.CK = clock  ;
cnt1.AR = !reset  ;
cnt1.OE = out_enable  ;

cnt2.D  = cnt2 & !cnt1 & !load
          # !cnt2 & cnt1 & cnt0 & !load
          # cnt2 & !cnt0 & !load
          # load & d1  ;
cnt2.CK = clock  ;
cnt2.AR = !reset  ;
cnt2.OE = out_enable  ;

cnt3.D  = cnt3 & !cnt2 & !load
          # cnt3 & !cnt1 & !load
          # !cnt3 & cnt2 & cnt1 & cnt0 & !load
          # cnt3 & !cnt0 & !load
          # load & d0  ;
cnt3.CK = clock  ;
cnt3.AR = !reset  ;
cnt3.OE = out_enable  ;

/* Terminal Count function */

tc.D  = cnt0 & cnt1 & cnt2 & cnt3  ;
tc.CK = clock  ;
tc.AR = !reset  ;
tc.AP = preset;

/* Buried Node Equations */

/* Counter load is the output of a 4-bit comparator comparing */
/* a[0..3] and b[0..3]  */
load = !( !a3 & b3  # a3 & !b3 # !a2 & b2
          # a2 & !b2 # !a1 & b1 # a1 & !b1
          # !a0 & b0 # a0 & !b0 ) ;

/* Multiple Product term Preset and Output Enables */

out_enable  = (en1 $ en2);

preset = pre1 $ pre2;
```

7.7 Hint 7- Using Slew Rate Control

The slew rate control on the the ATF1500A allows you to control the slew rate of each macrocell output. This feature allow you to customize the operation of the ATF1500A to the timing needs of your system. Slower switching outputs can reduce system noise. This is the default setting for all outputs by the fitter. If specific outputs in your design need to interface with higher speed devices in the system or you want to optimize your design for performance, you can set the each macrocell output individually to a fast slew rate.The following examples in Figure 7-7 and 7-8 respectively, illustrate how to specify the output slew rate in ABEL and CUPL.

```
module slewrate;
Title 'Slew Rate Control Example for the ATF1500A';

slewrate device 'p1500at';

library 'fit1500';

TURBO_OUTPUT(O1);   "The fitter will set outputs O1..O2 to a fast slew rate
TURBO_OUTPUT(O2);   "All other outputs will default to slow slew rate

i1,i2,i3 pin  ;

"Outputs

O1,O2,O3 pin ;
O4,O5    pin ;            " Output O3..O5 will be slow slew rate outputs

Equations.
.
```

Figure 7-7. Slew Rate Control - ABEL Example

A CUPL Example (Figure 7-8.) follows.

```
Name Slewrate;
Device f1500at;

$INCLUDE fit1500.m;

TURBO_OUTPUT(O1);    /* Specify outputs O1..O2 to be fast slew rate */
TURBO_OUTPUT(O2);

/* Inputs */
Pin = i1;
Pin = i2;
Pin = i3;

/* Outputs */
Pin = O1;
Pin = O2;
Pin = O3;
Pin = O4;
Pin = O5;
```

```
/* Logic Equations */
.
```

Figure 7-8. Slew Rate Control CUPL Example

7.8 Hint 8 -Using the Power Down Pin

The power down pin allows the user to externally power down the ATF1502AS to a zero power mode. The fitter defaults to having the power down pin disabled so it is available for logic. However, if power consumption is an important requirement in your design, enabling this pin allows the ATF1502AS to be powered down when your system requires it. Therefore, you can customize the power consumption of the ATF1502 to meet your specific system design needs. For example, the ATF1502 could be powered down when all inputs are idle, and then powered back up when you expect inputs to change. Please refer to Figures 3 and 4 for ABEL and CUPL examples on how to enable the power down pin.

Section 8

Appendix A - Error listing

The following is a list of errors and warning messages for the ATF1500 fitter applicable to the ATF1500A device along with example and explanations of their meaning.

ERROR- $signal references an unavailable pin number	Signal is assigned to VCC or GND or to a pin already defined in the design.
ERROR- $signal references a pin(node) Pin_Number that is not usable	Specific output signals or nodes assigned to a GLOBAL input. Two or more signals have the same pin assignments
ERROR- failed to assign $signal at $pin_number	Fitter was unable to assign specifed signal to specific pin number indicated in design file.
ERROR- there are $n pre-assignment problems	This error lists the pre-assigned pins which could not be assigned during the pre-assignment pass
ERROR- $signal could not be placed	Fitter is not able to place the specified floating signals during this design pass.
ERROR- Failed to place floating signals due to limited resources	This error is the same as above but means that all fitting passes were tried and the fitter could still not place the indicated floating signals.
ERROR- $signal .CE or .CK unable to use global	The defined signal was unable to be placed on the

clock pin which is already used.	Global clock GCLK because this input is already being used by another signal.
ERROR-Design has no inputs to DRC( )	There is no logic in the design file. The fitter has no logic to fit.
ERROR- $signal has wrong pin(node) preassignments $pin number	The specified pin or node is assigned an invalid pin or node number in the device.
ERROR- Design has too many output pins. $num used and $num available	Too many output pins are used by the design. Error indicate number used by the design and the actual number available on the ATF1500A.
ERROR- Design has too many input pins. $num used with $num input-only and $num I/O pin available	Too many input pins are used by the design. Error indicates number used by the design and the actual number available on the ATF1500A.
ERROR- Design has too many nodes, $num used with $num possible.	Design is using more foldback or buried register/combinatorial nodes than are available for this design. Error indicates number used by design and actual number available on the ATF1500A.
ERROR- registered $signal has combinatorial feedback	Signal indicated is using a combinatorial feedback from a register output, which in not available in the macrocell
ERROR- $signal has un-supported register type REG_JK	Signal indicated is using JK flip-flops which are not supported on the ATF1500A macrocell.
ERROR- $signal has unknown register type $type	The register type for the signal indicated in not supported on the ATF1500A.
ERROR- $signal has assigned multiple register types	Same signal is assigned two different register type such as D or T type in the logic. Only one type allowed.
ERROR- $signal uses extension .LH, only .LE is supported by CUPL	Specific signal in CUPL design is using the wrong suffix for level triggered latch output. Only .LE suffix is allowed.
ERROR- $signal uses extension .LE, only .LH is supported	Specific signal in ABEL design is using the wrong suffix for level triggered latch output. Only .LH suffix is allowed.
ERROR- $signal uses extension .SP only .AP is supported	Specific signal is using a synchronous preset product term which is not supported on the macrocell. Only asynchronous preset is allowed.
ERROR- $signal uses extension .J or .K which is not supported	Specific signal is using .J or .K suffix to define a JK flip-flop which is not supported on macrocell.
ERROR- $signal uses extension .FC which is not supported	Specific signal is using an invalid feedback extension which is not support on the macrocell

ERROR- $signal uses unknown extension	Specific signal is using an invalid extension
ERROR- must have a clock	The logic above this line is declared to be a registered output but no clock input is defined.
ERROR- Cannot assign both CLK and CE on product term.	The logic above this line is using both the .CLK and .CE input extension for the same logic function.
WARNING- No test vectors read	No test vectors are included in the source file.

Appendix B - ATF1500A Macro Library Listing

The following are detailed descriptions, and syntax for using the ATF1500A macros available in the macro library. To install these macros within your ABEL and CUPL environment please refer to the ABEL and CUPL Sections 4.4 of this manual. These macros are only applicable to the ATF1500A/AL/ABV device in PLCC44 and TQFP44 packages

☞ *Macros which enable specific signals have a higher priority than the same macro which enables the feature for all outputs or nodes. For example,*

FOLDBACK (X0 X1 X3); "FOLDBACK option *has higher priority than FOLDBACK_ALL*
FOLDBACK_ALL (OFF); "(even when FOLDBACK_ALL is OFF).

The above is true for all macros that have the global ON/OFF and individual signal setting options.
The individual setting option always has a higher priority than the global setting option.

Higher Priority	*Lower Priority*
GLOBAL	*GLOBAL_ALL*
CASCADE_LOGIC	*CASCADE_LOGIC_ALL*
TURBO_OUTPUT	*TURBO*
XOR_SYNTHESIS	*XOR_SYNTHESIS_ALL*

Below is the listing of the macros.

```
"*************************************************************************
EXP(A);   A = Node Labels
"
"  Implements specific buried COM signals on the foldback nodes in the device. Your "equations must be in
the following form.
"
"     foldback_node = (a & b & c &....);
"
"  In the ATF1500A device, the foldback nodes are a NAND function. With the EXP macro, the ! (not) is
"automatically assumed by the  ATF1500A Fitter.
"
"Note: If you are simulating the foldback nodes via the Simulate Equations, Simulate Optimized
"Equations or Simulate Fitted Equations ABEL command, then the vectors will fail simulation
"because ABEL simulates the equations as specified in the file and does not assume the invert condition.
"However, your JEDEC file will have the invert condition implemented.
"  E.g.
"     EXP (X0);
"
"     Equations
"     X0 = (!I1 & I0);   You must not use ! in the equation.
"
"   Refer to the FOLDBACK_LOGIC macro below for an alternate option.
```

"**

FOLDBACK_LOGIC(A); A = Node Labels

" This option is similar to the EXP option except that the ATF1500A Fitter will not assume the '!' in the
" AND function. (to get a NAND function) The '!' must be specifed in the foldback equations.
"
" Your equations must be in the following form.
"
"   foldback_node = !(a & b & c &....);
"
"In the ATF1500 device, the foldback nodes are NAND function. With the FOLDBACK_LOGIC macro, "you
must specify the '!' in the equations.
"
" Note: Unlike the EXP option, the equation simulation will pass with this fitter option.
"
" E.g.
"   FOLDBACK_LOGIC(X0);
"   Equations
"   X0 = !(!I1 & I0);   You must use the '!' in the equation.

"**

GLOBAL_ALL(A); A = ON/OFF
"Turns ON/OFF global input pin fitting assignments. Sets the most commonly used CLOCK, RESET or "OE
pins to the global pins. Note that the RESET and OE pins have to be Active Low or inverted in the ".AR and
.OE equations. The default condition is ON.
" E.g. #1
"
"   GLOBAL_ALL (ON);   This statement is not necessary because
"   Reset pin;   the GLOBAL default condition is ON.
"   OE pin;
"
"   Equations
"   OutReg.ar = !Reset; Assigned to the GCLR global pin.
"   OutReg.oe = !OE;   Assigned to the OE1 or OE2 global pin.
"
" E.g. #2
"
"   GLOBAL_ALL (ON);
"   !Reset pin;
"   !OE pin;
"
"   Equations
"   OutReg.ar = Reset;   Assigned to the GCLR global pin.
"   OutReg.oe = OE;   Assigned to the OE1 or OE2 global pins.

"**

GLOBAL(A); A = Input Pin Label
"
" Assigns the specified input(s) to the global input pins: GCLK, GCLR,
" OE1 and OE2 pinz.
" E.g.
"   GLOBAL(RESET);

"**

CASCADE_LOGIC_ALL(A); A = ON/OFF

"Turns ON/OFF cascade logic for fitter optimization. This option improves the "performance of the design,
but at the expense of device resources. The default condition is ON.
"

" E.g.
"   CASCADE_LOGIC_ALL (ON):

"***

CASCADE_LOGIC(A); A = Output Pin Label

"Allows specific outputs to use the cascade logic optimization for higher performance.
"

" E.g.
"   CASCADE_LOGIC (OUT); OUT to use cascade logic for logic implementation.

"***

SOFT_BUFFER_INSERTION(A); A = ON/OFF
"

" This option enables the fitter to create physical device nodes in the device, i.e. prevents the fitter from
"collapsing the combinatorial nodes. The default is OFF.
"

" Ex.
"   SOFT_BUFFER_INSERTION (ON); Turns ON physical node option

"***

SOFT(A); A = Node Labels
"

" This option assigns physical device nodes in the device, i.e. prevents the fitter from collapsing the
"combinatorial nodes.
"

" Ex.
"   SOFT(A); Assigns node A to be a physical device node

"***

TURBO(A); A = ON/OFF
"

" Sets all outputs to FAST (ON) or SLOW (OFF) slew-rate. Default condition is OFF.
" This is only applicable to the ATF1500A.
" E.g.
"   TURBO (OFF):

"***

TURBO_OUTPUT(A);   A = Output Pin Label
"

" Sets specific outputs for FAST slew-rate. This is only applicable to the ATF1500A.
"

" E.g.
"   TURBO_OUTPUT (OUT);

"***

XOR_SYNTHESIS_ALL(A);   A = ON/OFF
"

" Turns ON/OFF XOR synthesis for logic optimization. Default condition is ON.
"

" E.g.
"   XOR_SYNTHESIS_ALL (ON);

"**
XOR_SYNTHESIS(A);   A = Output Pin Label
"
" Sets specific outputs to use XOR synthesis.
"
" E.g.
"   XOR_SYNTHESIS (OUT);

"**
OPTIMIZE(A);   A = ON/OFF
"
" Turns ON/OFF all logic optimization features, including foldback and cascade logic (sharable and parallel expanders), and XOR synthesis optimization. The default condition is ON.
"
" E.g.
"   OPTIMIZE (ON);

"**
JEDEC_FILE(A);   A = JEDEC Filename
" Allows a different file name for the generated JEDEC file. The default file name is the filename "specified in the DEVICE statement in the ABEL file.

"**
VECTOR_FILE(A);   A = .TMV Filename
" Allows another ABEL vector file (.TMV) to be read by the fitter.

"**
SLEEP(A);   A = ON/OFF
" Sets the Power Down mode for the ATF1500A device only.

"**