AT-Editor User Manual

Stefan H.-M. Ludwig
ludwig@inf.ethz.ch

Abstract

The AT-Editor is a program package for the graphical design of circuits for the Atmel AT6000 Field Programmable Gate Array architecture. The functionality of the software supports the fast design of circuits on the computer screen, downloading of the circuit onto the hardware of an AT-Board, and testing. Parts of designs can be copied into other designs, thereby allowing the construction of libraries of tested components. Before reading this manual, the user should become acquainted with the AT6000 architecture.

Necessary Modules and Files

The editor package consists of the following modules:

ATGAs	Data structures representing designs, loading and storing of designs
ATLoader	Loader for the AT-Board
ATFramesD	Display procedures used by the editor
ATFrames	Editing operations
AT	Command module and programming interface for the AT-Board
AT.Tool	Tool text with explanations
AT.Menu.Text	Text displayed in the menu frame
AT.Configuration.Text	Text defining default labels and the pin structure for various chip sizes
AT.Manual.Editor.Text	This documentation
AT.Scn.Fnt	Patterns for the editor
Latch.At	Example design

AT.Tool contains the most often used commands with short explanations. *AT.Menu.Text* can be altered by the user to suit his or her needs. If it is not present, a default menu is used.

Opening, Storing, and Loading Designs

The command *AT.Open ["ATchip.pins"] Name.At* loads a design from disk and opens a viewer with a graphical representation of the design. The string *"ATchip.pins"* specifies the dimensions and package of the chip for a new design. If no chip type is specified, *"AT6002.84"* is used. File names of designs should end on *At* (convention). Like with other Oberon commands, an arrow (↑) indicates the last text selection to be the argument of a command (*AT.Open* ↑).

Important note to former users of CLi: The Atmel architecture has a different local bus interface than the former Concurrent Logic FPGA. When loading an old Cli design, possible warning messages with cell positions are written to the log. These positions mark cells, which use the local bus in way that is incompatible with the Atmel FPGA. Open such a design with the old CL-Editor and compare the marked cells, making adjustments in the Atmel design where necessary.

If a design gets changed, an exclamation mark in the menu frame of the corresponding viewer indicates this fact. This way, all viewers showing modified designs are clearly marked.

Command *AT.Store* in the menu frame stores a design to a file using the name shown in the menu frame. A backup of the old design is kept on disk under the name *Name.At.Bak*.
Hint: To rename a design, one can simply edit the name in the menu frame with the mouse (delete the

name in the menu frame using the mouse and type the new name on the keyboard). After storing the design, it is available under the changed name.

AT.Load, another command in the menu frame, downloads the displayed design to the AT-Board and notifies the user through a message in the Log viewer (*loading done/not done*). If the design contains errors, a message with the coordinates of the erroneous cell is printed (–> *AT.Locate*).

Editing: General Remarks

The editor is used consistently with the mouse. Mouse button assignments and their functions were kept compatible with the normal Oberon text operations as closely as possible. The command *AT.Cells* in the menu frame toggles between two display modes. Either all cells are shown – even the ones not used – or only those cells, which actually have a content or get used for routing a bus. Furthermore, only those local and express buses are shown, which get used by a cell or are connected together via a repeater.

Canceling and Undo

By pressing *all three buttons at the same time*, a started operation can be canceled. The command *AT.Undo* in the menu frame can be used to undo the last editing operation. Undo is involutory, i.e. invoking it a second time redoes the operation.

Usage of Mouse Buttons

The *left button* is used to edit cells, pads, and repeaters. In most cases, a menu will appear from which a certain item can be chosen. By leaving the menu with the mouse and releasing the button, the operation is canceled and nothing is changed.

The *middle button* is used to shift the view of a design or to move or copy a cell selection.

The *right button* is used to select a cell stretch, a pad, or a repeater.

Editing Cells

Cells are edited with the left mouse button. Depending on the location where the button is pressed, different actions take place: Either the inputs or outputs of a cell are edited or the cell's content itself. Figure 1 below shows the sensitive areas of a cell. By pressing the left button in the respective area, the involved states get changed:

A/B inputs: The input is turned on or off (*toggled*). Depending on the selection of inputs the picture inside the cell changes as not selecting an input results in a "1" being supplied at that input. The inputs can be toggled on all four sides of the cell. Near the A output (fat terminal ⊔) lies the B input and near the B output (thin terminal ⊔) the A input. The lower outputs and the upper inputs correspond graphically with the patterns inside the cell. If other directions are chosen, one has to imagine the signal flow since the pattern is not rotated accordingly.

L-bus input/output: If the cell *reads* a local bus (Control, MuxAB) (middle right and middle left cell in Fig. 1) the arrow is drawn from the L-bus to the cell. The mouse has to be moved *from the bus to the cell* to enable a read. If the cell *writes* the bus (WriteL, Tristate) (middle cell in Fig. 1) the arrow is drawn from the cell to the L-bus. The mouse has to be moved *from the cell to the bus* to enable a write. The north-west and south-east buses form a pair, and each pair can be involved in *one bus operation only* (read/write/corner turn).

L-bus corner turn: A cell can implement one of two corner turns (lower left and upper left cell in Fig. 1). By clicking into the upper left or the lower right corner, the turn can be toggled. These connections are bi-directional and no arrow is drawn therefore.

It is possible to have the cell perform a corner turn without choosing the cell's content. This can be seen in the lower left cell in Fig. 1. The corresponding turn at the cell is chosen without first giving the cell a content.

Labels for A and B outputs: The *last text selection* can be copied to the lower A or B output by additionally pressing the middle button (left/middle interclick). This is analogous to *Set Caret/Copy* with normal texts. With left/right interclick the label at that position is cleared. If a label exists in a design already, an error message is printed to the Log. Labels may end with a single quote (') to indicate signals with negative logic. *AT.LabelA*, *AT.LabelB*, *AT.DeleteA*, *AT.DeleteB* are used to label cell stretches (see below).

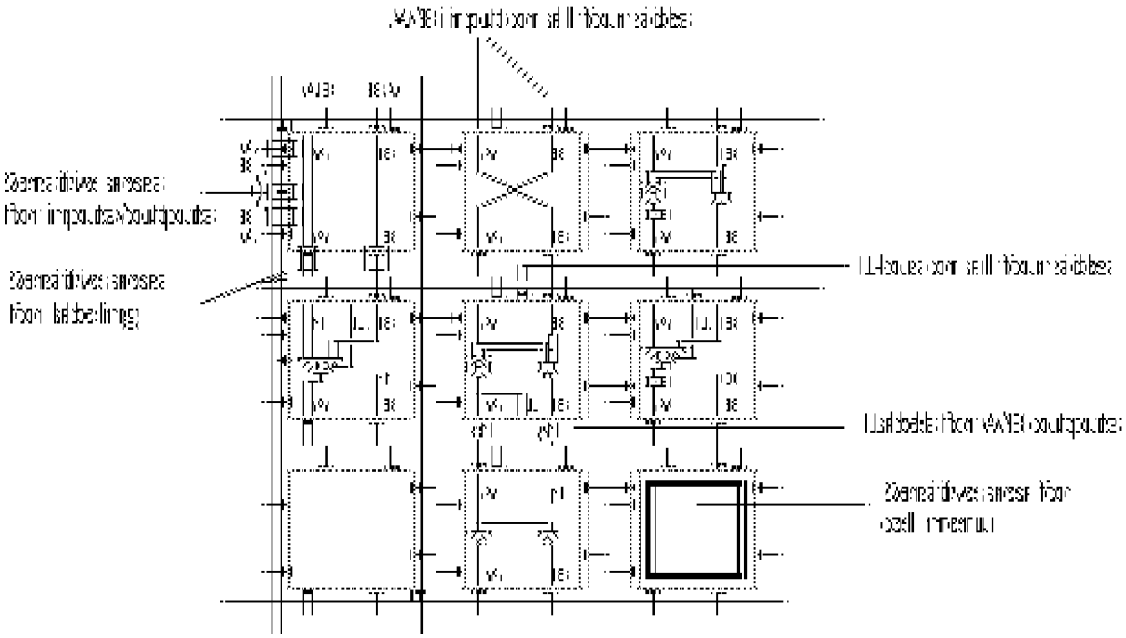

Fig. 1

Cell menu: When pressing the left button, a popup menu appears on the screen (Figure 2). If a cell has a content already, it will be highlighted with a frame. The first four items in the lower row constitute the *State* and the upper row constitutes the *In* and *Out* modes of a cell. The two Muxes on the lower right are shortcuts for state Xor/Reg and input Mux. If the left button is released *outside the menu*, no selection takes place.

By interclicking the *middle button* during the menu selection, the cell can be initialized (no routing, no state, no input, no output).

The content of the cell is drawn depending on which modes and state is chosen and which inputs are active. E.g. the Xor and Nand symbols in the middle cell of Figure 1 become a Not if input B is not active (A XOR 1 = ~A).

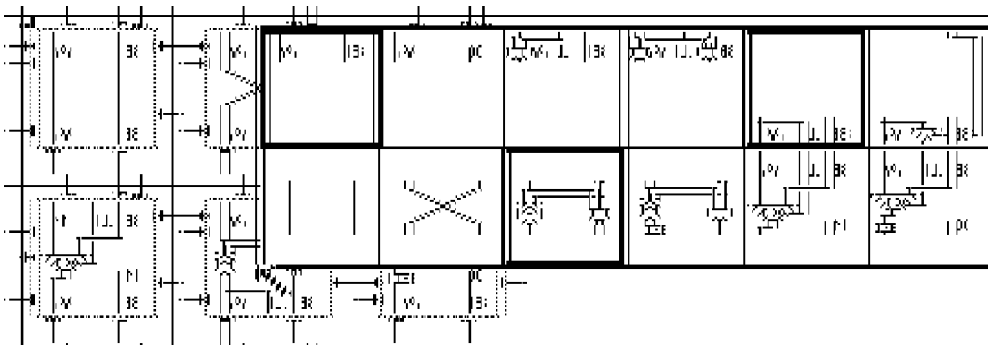

Fig. 2

Editing Pads

The pads, too, can be edited with the left mouse button. A menu appears if the button is pressed inside a pad (Figure 3). The function for the tri-state gate can then be chosen. The current state of the pad is highlighted with a frame. If the left button is released *outside the menu*, no selection takes place.

By interclicking the *middle button* during the menu selection, the pad can be initialized (0, no output).

1. item (left in menu in Fig. 3): The pad acts as input only. "0" is drawn inside the pad box (output off). This is the most secure setup of a pad since no signals can leave the chip.

2. item: The pad is controled by the L-bus *running perpendicular to the pad's side* (the vertical L-bus to the right (east) of the output cell in the figure). This state is indicated with a bar "|", which is perpendicular to the pad's side.

3. item: The pad is controled by the L-bus *running parallel to the pad's side*. A dash "–" parallel to the pad's side is shown in this state, which is the normal case for data pads (D.0..D.7) on the left side of the chip on the AT-Board.

4. item (right): *The signal of the output cell drives the pad always*, which is indicated by a "1". Care must be taken with this state: Only the address lines (a.0..a.14) on the top of the chip on the AT-Board should have this state. For all other pads this state should not be selected, as external logic or the chip itself could be destroyed.

Labels can be set at pads, too – application is analogous to that of cells. The sensitive area for pads is at the input or output. Labels are drawn above or below a pad (cf. Figure 3: *sig*).

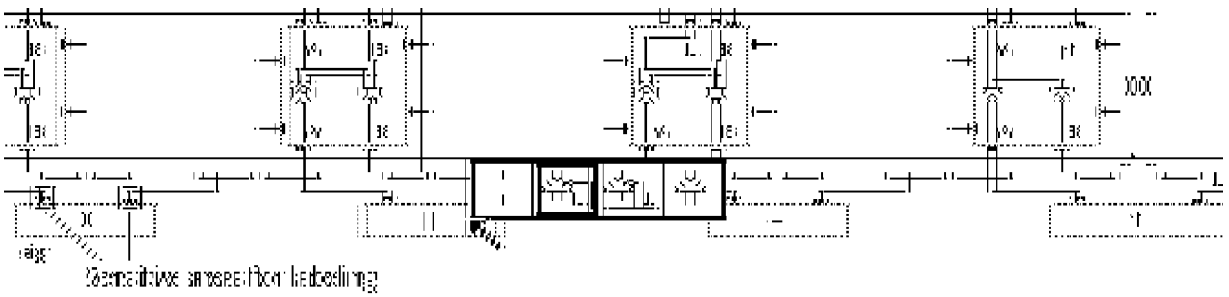

Fig. 3

Editing Repeaters

The state of repeaters can be chosen with a menu (Figure 4). All possible connections between local and express buses are presented in the menu. The current state is highlighted with a frame. If the left button is released *outside the menu*, no selection takes place.

By interclicking the *middle button* during the menu selection, the repeater can be initialized (no connections).

The two last columns in the menu in Fig. 4 call for an explanation:

1: L-bus and E-bus are crossed over (E–>L L–>E)
2: L-bus writes to L- and E-bus or vice versa (L–>E and L–>L or E–>E and E–>L)

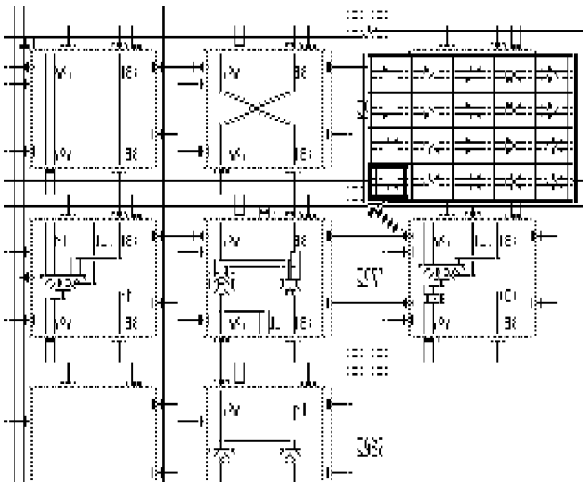

Fig. 4

Selection

With the right mouse button, multiple cells and single pads and repeaters can be selected. Certain commands require cells, pads, or repeaters as arguments and these are specified by the last selection.

Single cells, *pads*, and *repeaters* can be selected by simply pressing the *right button* inside the menu area. If *multiple cells* should be selected, the mouse should be brought to the lower left cell, the right button should be depressed and *held down*, and the mouse should then be dragged to the upper right cell one wishes to select. This way, a rectangular area can be selected even over repeater boundaries.

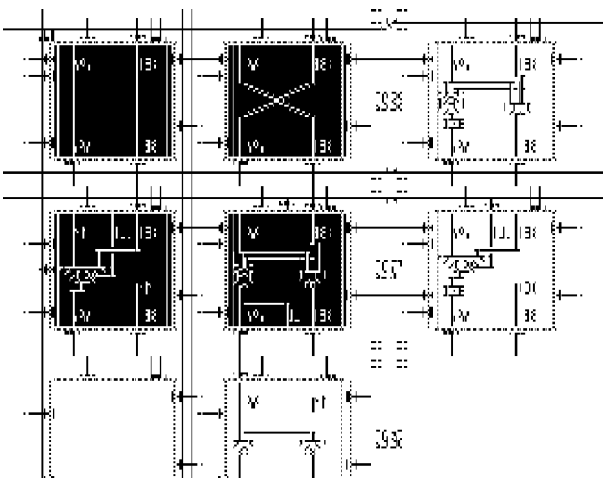

Fig. 5

Shifting the View, Using Thumbnail, Moving and Copying Cells

Pressing the *middle button of the mouse over a cell* and then moving the mouse will *shift* the visible portion of the design. The location where the button is depressed appears at the location where the button is released.
Pressing the *middle button between cells* brings up a *"thumbnail" view* of the design. Used cells can be seen as small dots and the visible portion of the design is represented by a rectangle. Moving the rectangle to a new location will shift the view to that area.

If the middle and right buttons are depressed (*middle/right interclick*) the most recently selected cells are *copied* to the location where the buttons are released. Care should be taken in releasing the buttons only over a cell. If a rectangular area should be copied, the target cell must be the *lower left* cell. If the area cannot be copied as a whole, the operation is canceled.

If the middle and left buttons are depressed (*middle/left interclick*) the most recently selected cells are *moved* to the location where the buttons are released, i.e. they are deleted at the old location. Labels at outputs are moved as well.

During these operations (moving or copying), repeaters are not moved or copied along. It is possible, though, to move or copy cells into another viewer. This way, designs from one viewer can be copied into another. Attention: *AT.Undo* has to be invoked in each viewer where a modification occurred.

List of Mouse Button Combinations

left button	menu for cell/pad/repeater
	(release outside menu to cancel)
	toggle A/B inputs
	toggle L-bus input/output
middle (interclick)	initialize cell/pad/repeater if inside menu
	copy text selection to label
right (interclick)	delete label
middle button	shift view, thumbnail view
left (interclick)	move cell selection (incl. labels)
right (interclick)	copy cell selection
right button	select
all three buttons	cancel started operation

List of Commands

Commands in the menu frame

AT.Locate	cf. *Locate* below
AT.Array	cf. *Array* below
AT.Cells	show all cells / only used cells
AT.Load	download design to the AT-Board
AT.Check	check consistency of design for downloading, but don't download
AT.Undo	undo/redo the last operation
AT.Store	store design to file using name in the menu frame

Commands in AT.Tool

Many commands in *AT.Tool* take the marked ("*") viewer as an argument. If no viewer is marked and the command is not in the menu frame, the viewer containing the most recent selection is used. With this, the star does not have to be setup every time when a selection exists already. *Viewer* written in italics indicates if a command expects this kind of viewer.

AT.Open ["ATchip.pins"] Name.At	open a viewer displaying the design stored in file *Name.At*
AT.Open ["ATchip.pins"] ↑	the last text selection is taken as argument

Commands *Locate* and *Array* are used to find (arrays of) cells by means of their names and coordinates.

AT.Locate name	*name* is searched for as a label in *Viewer* and the corresponding cell or pad is selected
AT.Locate x y | pos	the cell or pad at *x y* or *pos* in *Viewer* is selected Possible values for *pos* are: *LowerLeft, UpperLeft, LowerRight, UpperRight*
AT.Locate ↑	the last text selection is taken as argument
AT.Locate	in the menu frame is the same as *AT.Locate ↑*
AT.Array prefix	labels starting with *prefix* are searched for in *Viewer* and the corresponding cells are selected (used for locating arrays)
AT.Array ↑	the last text selection is taken as argument
AT.Array	in the menu frame is the same as *AT.Array ↑*

SetClock and *SetReset* are used to set the value of the clock/reset signals in the selected column(s). It is important that the top-most row is selected for setting the clock, and the bottom-most (0.) row for setting the reset signal of a column. The command is applied to *all selected* columns. Note: The reset signal is active low ("0"). *SetPassGate* is used to turn the pass-gate of an east or south repeater on or off (to the right of or below a cell).

AT.SetClock Aout	set Clock of column(s) to A-Output of the top-most cell in the column(s)
AT.SetClock Global	set Clock of column(s) to global clock signal (Ceres-Clock)
AT.SetClock Express	set Clock of column(s) to express bus south of top-most row
AT.SetClock Off	turn Clock of column(s) off
AT.SetClock ↑	the last text selection is taken as argument
AT.SetReset Aout	set Reset of column(s) to A-Output of the bottom-most cell in the column(s)
AT.SetReset Global	set Reset of column(s) to global reset signal
AT.SetReset Express	set Reset of column(s) to express bus north of bottom-most row

AT.SetReset Off	turn Reset of column(s) off
AT.SetReset ↑	the last text selection is taken as argument
AT.SetPassGate On	turn pass-gate of selected east or south repeater on
AT.SetPassGate Off	turn pass-gate of selected east or south repeater off
AT.SetPassGate ↑	the last text selection is taken as argument
AT.Reset	all flip-flops in the chip are reset (global reset)

Information about a design is written to the Log with:

AT.Inspect ↑	detailed information about the selection (cell/pad/repeater)
AT.Clocks	clock assignments for all columns in *Viewer*
AT.Resets	reset assignments for all columns in *Viewer*
AT.Labels	used labels in *Viewer*
AT.Statistics	number of used cells and buses in *Viewer*

The commands below allow for automatic labeling of cell stretches. The following must hold for all label commands: *(dx # 0) and (dy = 0) or (dy # 0) and (dx = 0)*

AT.LabelA "label" dx dy [beg]	label A-output of selected cells
AT.LabelB "label" dx dy [beg]	label B-output of selected cells
AT.DeleteA dx dy	delete label at A-output of selected cells
AT.DeleteB dx dy	delete label at B-output of selected cells
AT.Label/DeleteA/B ↑	the last text selection is taken as argument

The label must be in quotes and contain a "*" character; the "*" is replaced by a range of numbers specified by the remaining parameters. A negative *dx* or *dy* is used to decrement instead of incrementing the label values, but the cell indices are always incremented by ABS(*dx*) and ABS(*dy*). The label values are always between 0 and 9. Values larger than 9 must be entered as shown in the next to the last example:

AT.LabelA "x." 0 2*	will label the A-output of every second vertical cell with x.0, x.1, x.2, ...
AT.LabelB "M..5" 0 1 3*	will label the B-output of every vertical cell with M.3.5, M.4.5, M.5.5, ...
AT.LabelA "y." 0 –1 9*	will label the A-output of every vertical cell with y.9, y.8, y.7, ...
AT.LabelB "z.1" 3 0*	will label the B-output of every third horizontal cell with z.10, z.11, z.12, ...
AT.DeleteA 0 2	deletes labels at A-outputs of every second vertical cell

A bus may be labeled using *AT.LabelBus*. A cell near the bus has to be selected. The label for the corresponding bus is drawn at the left or lower edge of an 8x8 block.

AT.LabelBus bus label	will put *label* to the *bus* near the selected cell
AT.LabelBus bus ↑	the last text selection is taken as argument for *label*

Possible values for *bus* are: *LBusN*, *LBusS*, *LBusE*, *LBusW*, *EBusN*, *EBusS*, *LBusE*, *LBusW*

Commands *Put* and *Get* are used to read and write the 4 ports of the AT-Board. Procedures *PutInt* and *GetInt* are the programming interface to the board.

AT.Put [port] val	write *val* to AT-port *port* (0 default) *(0 <= val <= 255, 0 <= port <= 3)*
AT.Put ↑	the last text selection is taken as argument
AT.Get [port]	read a byte from AT-port *port* (0 default) and write it to the Log *(0 <= port <= 3)*
AT.Get ↑	the last text selection is taken as argument

AT.PutInt(port, val: INTEGER); Write *val* to AT-port *port* *(0 <= val <= 255, 0 <= port <= 3)*
AT.GetInt(port: INTEGER): INTEGER; Read a byte from AT-port *port* *(0 <= port <= 3)*

With the command *SetColor* the different colors of the designs can be changed.

AT.SetColor kind col change the color of *kind* to *col*
(kind: Cell, Conn, LBus, EBus, Pad, Label, Menu, MenuHigh; 0 <= col <= 15)
AT.SetColor ↑ the last text selection is taken as argument

The following commands are used to setup the data file and the default chip. For more information about the data file and chip data refer to the default data file.

AT.ShowChip shows the name of the chip in the configuration file used as default
AT.SetChip chip sets the new name of the chip in the data file *(default: "AT6002.84")*