Rules for Reduced Complexity Block Diagrams

Purpose: To define a methodology for constructing block diagrams of reduced complexity for the top-level datapath in RTL design of authenticated ciphers.

Goals: Uniformity, simplicity, completeness, “shrinkability,” reduced time in construction and alteration.

Applicability: This methodology is applicable to the top level of an RTL block diagram for datapaths (particularly in authenticated ciphers), where the block diagram is intended to be rendered in a format suitable for academic, journalistic, or peer review publications and media. Examples include IEEE split-page format, Lecture Notes on Computer Science (LNCS) format, conference / lecture presentations in Microsoft Power Point, web-page or email embedded images. This format provides less information than a fully-developed block diagram; therefore, it does not alter the rationale for fully developed block diagrams.

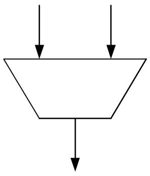
This methodology is especially applicable to authenticated ciphers. Authenticated ciphers typically contain complex control and switching mechanisms at the top level of the datapath. If rendered in full detail, top level block diagrams rapidly become cluttered, unmanageable, and cease to provide useful information to the reader.

This format is not applicable to ASM Charts, Interface Diagrams, lower-level block diagrams where detail is desired and clutter is not a concern, and non-RTL (i.e., logic gate, tabular, or circuit) diagrams.

Rules:

1. Multiplexers should be labeled without SEL lines and without internal selectors such as “1”, “0”, “001”, etc.

Example:



Rationale: Clutter reduction; assignment of selector values “0”, “1”, etc. is typically arbitrary and does not convey substantial information to the reviewer.

2. Registers:

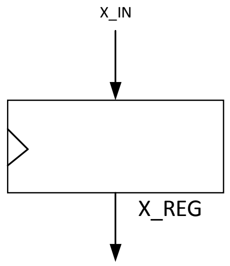
a. Basic control signals, such as “CLK”, “RST”, “CLR”, or “INIT” should not be labeled unless absolutely necessary to understand functionality.

b. Label registers using a triangular "wedge" to show clock, and label the register at the bottom right (*) of the register (i.e., not inside). No other labeling should be included inside the registers.

* The label can be placed at the bottom left, or in a convenient logical place, if it will not fit at the bottom right of the box.

Rationale: Provide sufficient information while reducing clutter in and around the register.

Example:

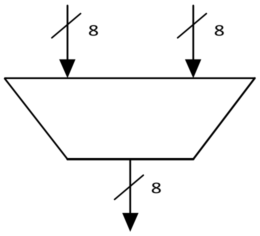


3. Explicitly label all bus widths, however, generic statements such as "all bus widths 128 bits unless indicated" are allowed and encouraged.

a. Use a bold line font for signals with bus widths greater than 1.

b. Use a thin line font for single bit signals.

Example:



Rationale: Clutter reduction.

5. Label lower level entities (such as "AES Enc") inside the box. The key external ports of the lower level entity should be displayed within the box, when practical. However, if so doing will present a cluttered appearance they can be omitted.

6. Use the following standard symbology as shown below (additional symbols are possible).

⊕ XOR Gate $\boxed{==}$ Comparator $\boxed{>>}$ Shifter $\boxed{>>>}$ Rotator $\boxed{+1}$ Incrementer

7. Do not use the generic wedge-shaped ALU symbol (i.e., the symbol often found in microprocessor block diagrams), unless absolutely necessary.

Rationale: Generic ALU functions are rare in cryptographic implementations, and may confuse the reader.

8. Avoid the use of logic gates, unless absolutely necessary to understand functionality.

Rationale: Logic gates should generally be avoided in RTL design; Clutter reduction.

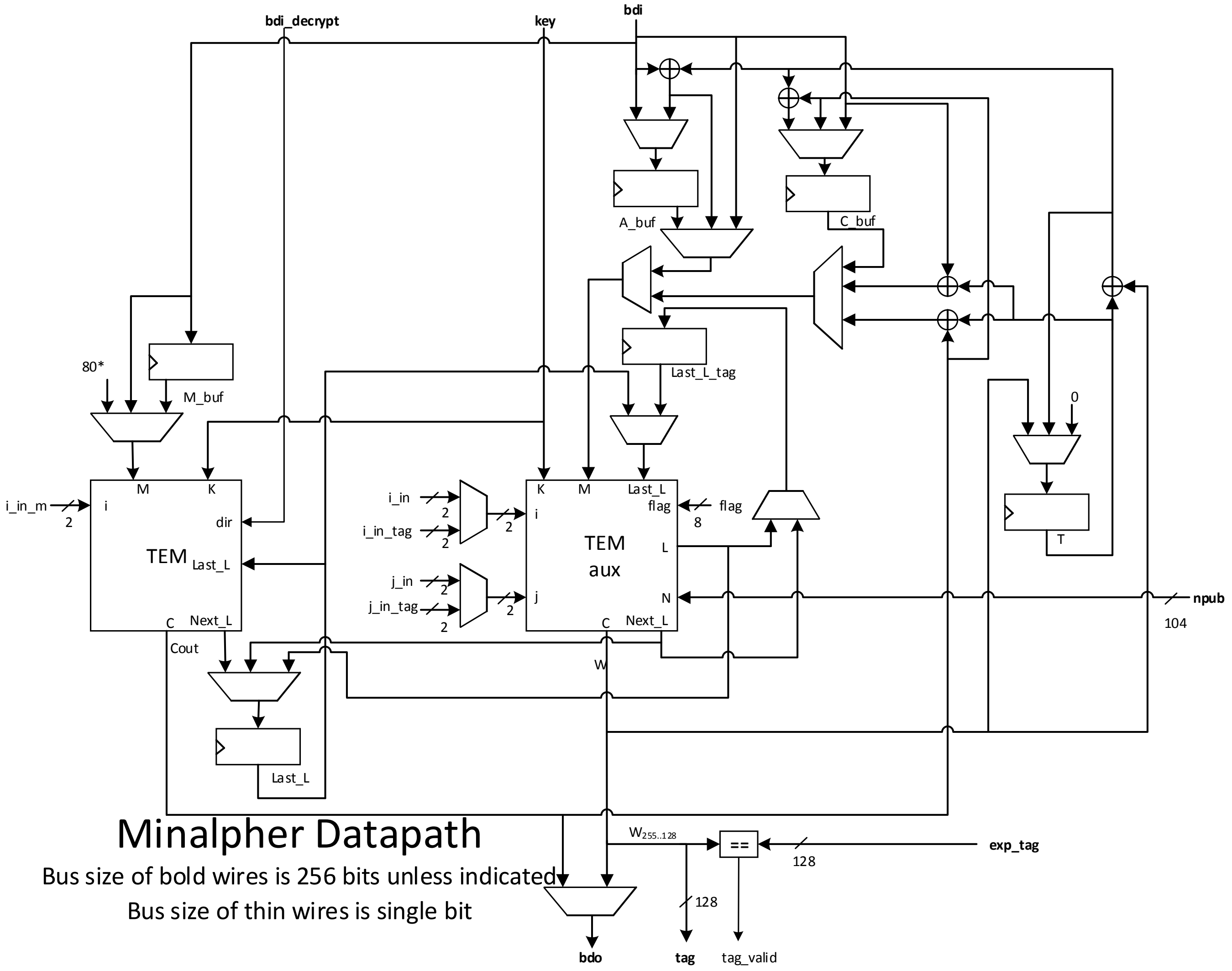
9. Generally avoid labeling intermediate signals where it is possible for a reader to understand the flow by just following along on the diagram.

10. Use either “flyovers” to indicate that wires do not touch, or “solder points” (i.e., dots) to show that connections exist. Note that xfig does not support flyover wires.

Rationale: Ensure that the reader understands where connections exist.

11. Signal names which specify a formal external port defined in the interface should be **bold font.**

The following block diagram is an example of a top level block diagram drawn using these rules, including flyovers:

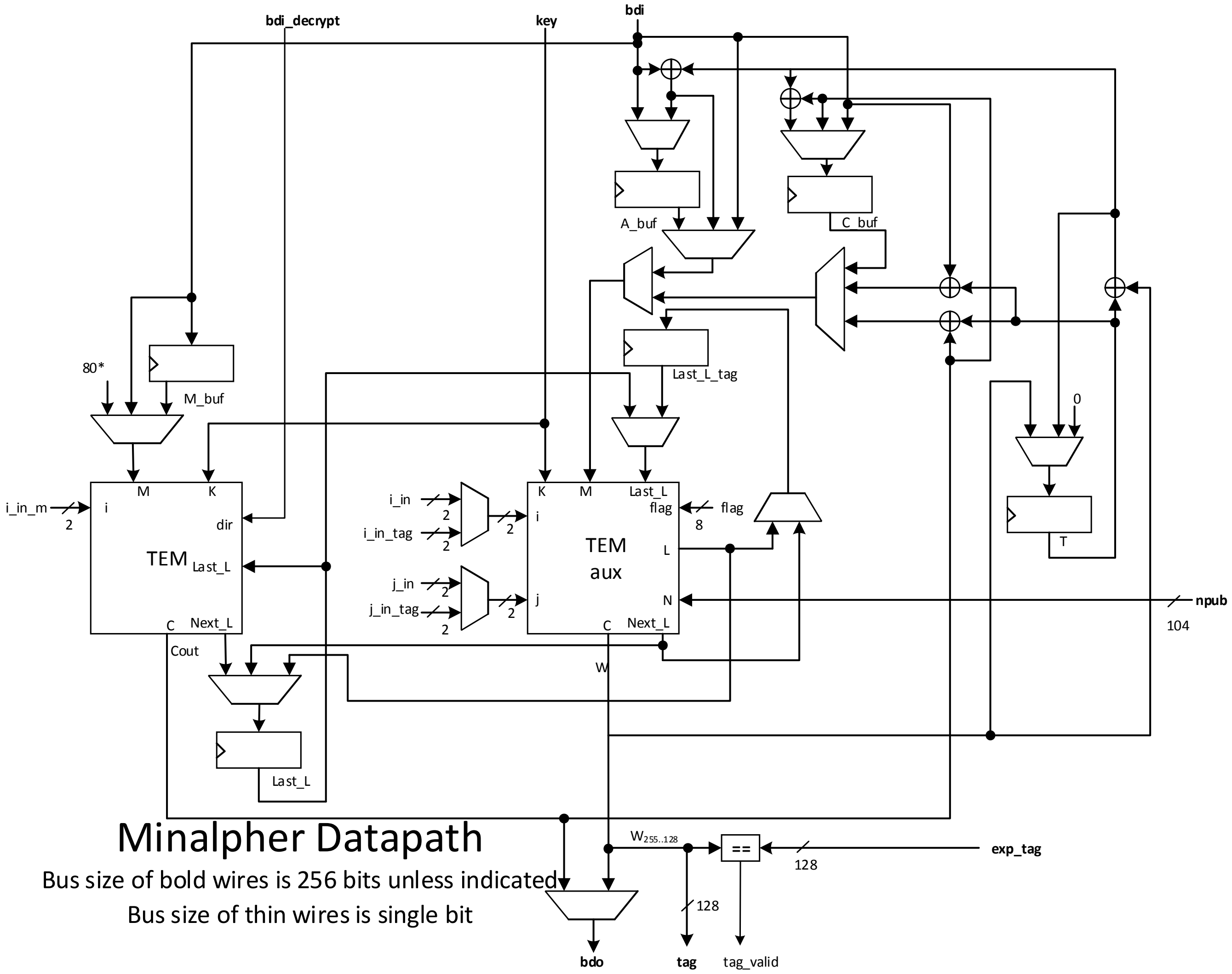


Minalpher Datapath

Bus size of bold wires is 256 bits unless indicated

Bus size of thin wires is single bit

The following block diagram is an example of a top level block diagram drawn using these rules, including solder points:



Minalpher Datapath

Bus size of bold wires is 256 bits unless indicated

Bus size of thin wires is single bit