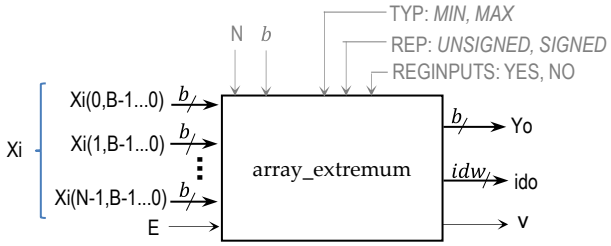
Array Extrema

COMPUTING AN EXTREMUM (MINIMUM OR MAXIMUM)

FULLY PIPELINED CIRCUIT: INTERALLY-GENERATED INDICES

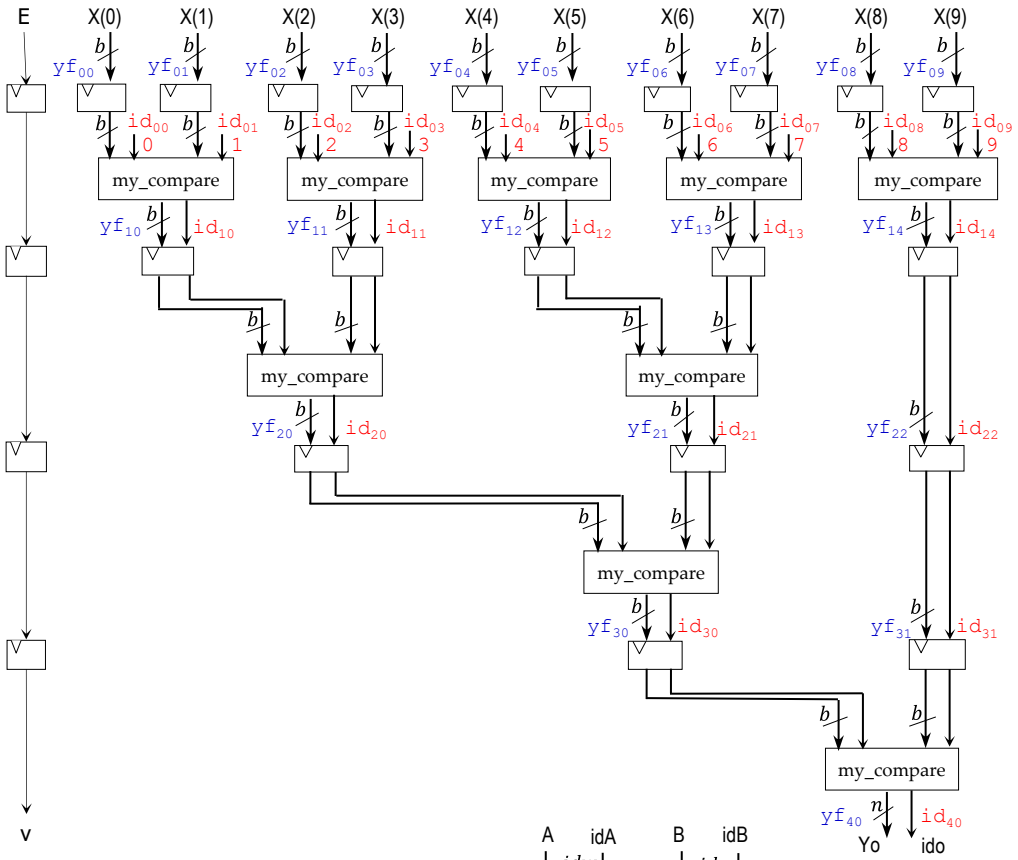
- This fully pipelined circuit, called 'array_extremum', determines the maximum or minimum of an n-element integer array.
- The pipelined architecture resembles that of the pipelined adder tree.

- The block diagram depicts the I/Os and parameters of the circuit.
- Parameters:
  - ✓ N: number of input (integer) elements
  - ✓ B: bit-width of each element
  - ✓ TYP: whether to compute maximum or minimum
  - ✓ REP: whether to treat input elements as unsigned or signed.
  - ✓ REGINPUTS: whether to have registered inputs or not.



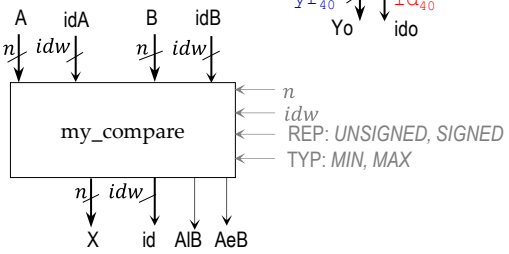
- Typically, the index of the extremum (maximum, minimum) is as important as the extremum itself. Thus, the circuit generates the extremum (Yo) and the respective index (ido).
  - ✓ The indices are generated internally, and it follows the convention from 0 to N-1. The bit-width is $idw = \lceil \log_2 N \rceil$.
  - ✓ The input indices (0 to N-1) do not need to go through input registers, as they are constants.

- Latency:
  - ✓ REGINPUTS=YES: $Latency = LV = \lceil \log_2 N \rceil$.
  - ✓ REGINPUTS=NO: $Latency = LV - 1 = \lceil \log_2 N \rceil - 1$. Here, the first row of registers does not exist.



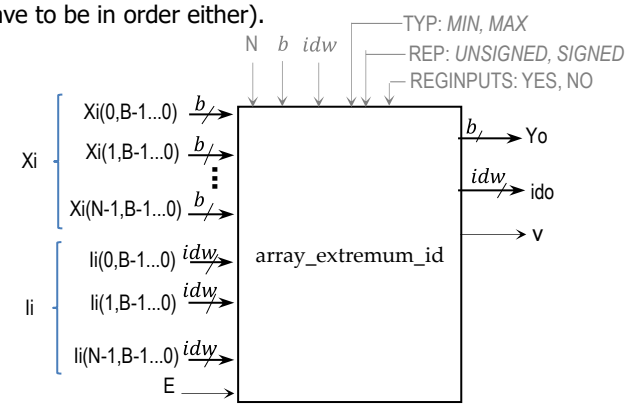
Comparator:

- This block, based on a 2C subtractor, determines either the maximum or the minimum value. It also computes:
  - ✓ The index of the extremum. The indices are fed externally.
  - ✓ AlB (A < B ?), and AeB (A = B?).



FULLY PIPELINED CIRCUIT: EXTERNALLY-FED INDICES

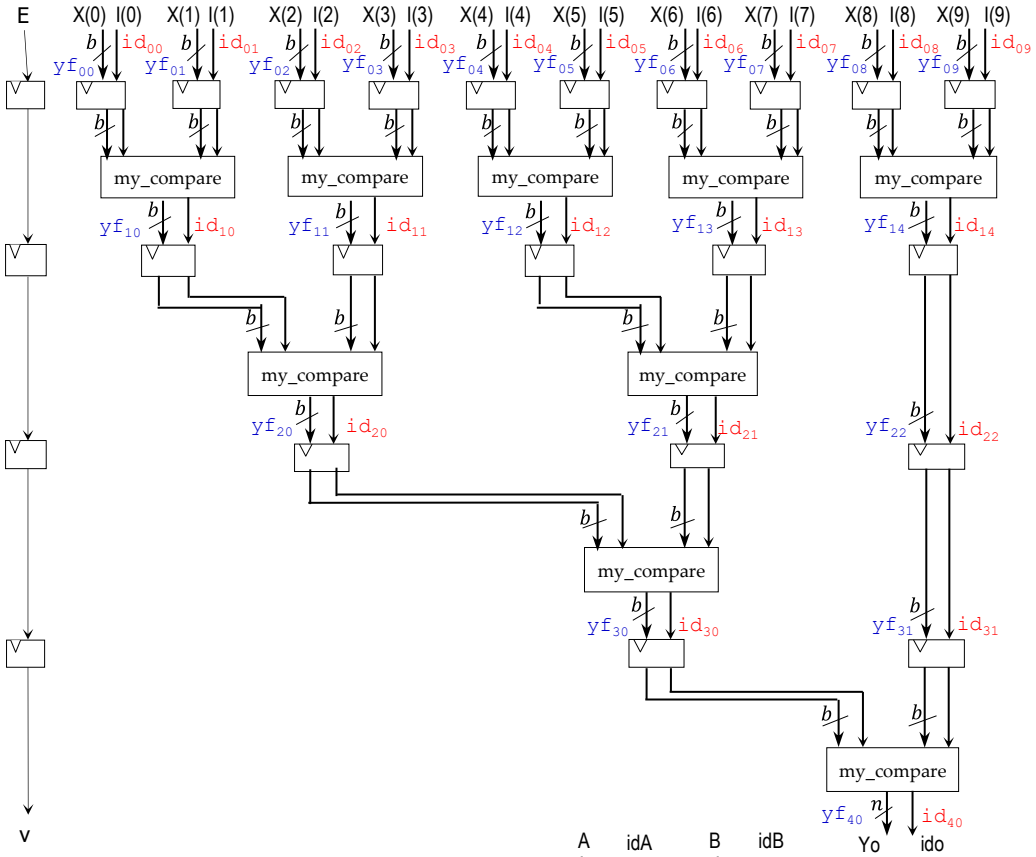
- This fully pipelined circuit, called '`array_extremum_id`', determines the maximum or minimum of an n-element integer array.
- The pipelined architecture resembles that of the pipelined adder tree.
- This circuit is nearly identical to that of '`array_extremum_id`', except that the indices are fed externally.
  - ✓ The bit-width of the indices usually is $idw = \lceil \log_2 N \rceil$. However, in some cases, it might be useful to consider larger bit-widths as the indices can come from a larger circuit (they don't have to be in order either).
  - ✓ Thus, the bit-width of the indices is selectable by the user.



- The block diagram depicts the I/Os and parameters of the circuit.
- Parameters:
  - ✓ `N`: number of input (integer) elements
  - ✓ `B`: bit-width of each element
  - ✓ `IDW`: bit-width of the indices
  - ✓ `TYP`: whether to compute maximum or minimum
  - ✓ `REP`: whether to treat input elements as unsigned or signed.
  - ✓ `REGINPUTS`: whether to have registered inputs or not.

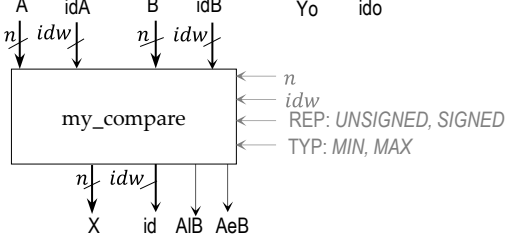
- Typically, the index of the extremum (maximum, minimum) is as important as the extremum itself. Thus, the circuit generates the extremum (`Yo`) and the respective index (`ido`).
  - ✓ The indices are fed externally (as an array). Thus, if `REGINPUTS=YES`, they go through input registers.

- Latency:
  - ✓ `REGINPUTS=YES`: $Latency = LV = \lceil \log_2 N \rceil$.
  - ✓ `REGINPUTS=NO`: $Latency = LV - 1 = \lceil \log_2 N \rceil - 1$. Here, the first row of registers does not exist.



Comparator:

- This block, based on a 2C subtractor, determines either the maximum or the minimum value. It also computes:
  - ✓ The index of the extremum. The indices are fed externally.
  - ✓ `AlB` (A < B ?), and `AeB` (A = B?).

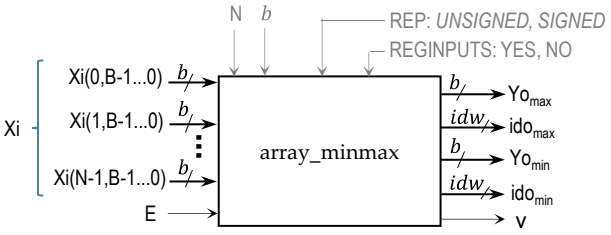


COMPUTING THE MINIMUM AND MAXIMUM

FULLY PIPELINED CIRCUIT: INDICES GENERATED INTERNALLY

- This fully pipelined circuit, called 'array_minmax', determines the maximum and minimum of an array of n integer elements.

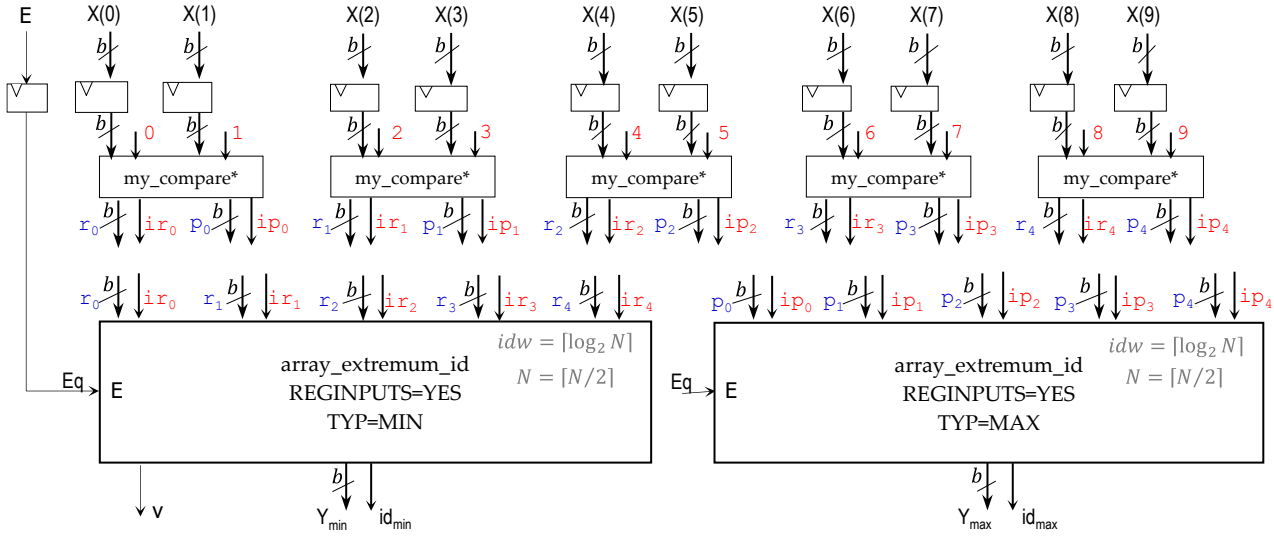
- The block diagram depicts the I/Os and parameters of the circuit.
- Parameters:
  - ✓ N: number of input (integer) elements
  - ✓ B: bit-width of each element
  - ✓ REP: whether to treat input elements as unsigned or signed.
  - ✓ REGINPUTS: whether to have registered inputs or not.



- Typically, the indices of the maximum and minimum are as important as the values themselves. Thus, the circuit generates the extrema (Yo_{max}, Yo_{min}) and the respective indices (ido_{max}, ido_{min}).
  - ✓ The indices are generated internally, and it follows the convention from 0 to N-1. The bit-width is $idw = \lceil \log_2 N \rceil$.
  - ✓ The input indices (0 to N-1) do not need to go through input registers, as they are constants.

- Latency:
  - ✓ REGINPUTS=YES: $Latency = LV = \lceil \log_2 N \rceil$.
  - ✓ REGINPUTS=NO: $Latency = LV - 1 = \lceil \log_2 N \rceil - 1$. Here, the first row of registers does not exist.

- We utilize the 'array_extremum_id' block. Note the values provided for the parameters: $idw = \lceil \log_2 N \rceil$, $N = \lceil N/2 \rceil$.



Comparator*:

- This block, based on a 2C subtractor, determines the maximum and the minimum value. It also computes:
  - ✓ The indices of the extrema. The indices are fed externally.
  - ✓ AlB (A < B ?), and AeB (A = B?).