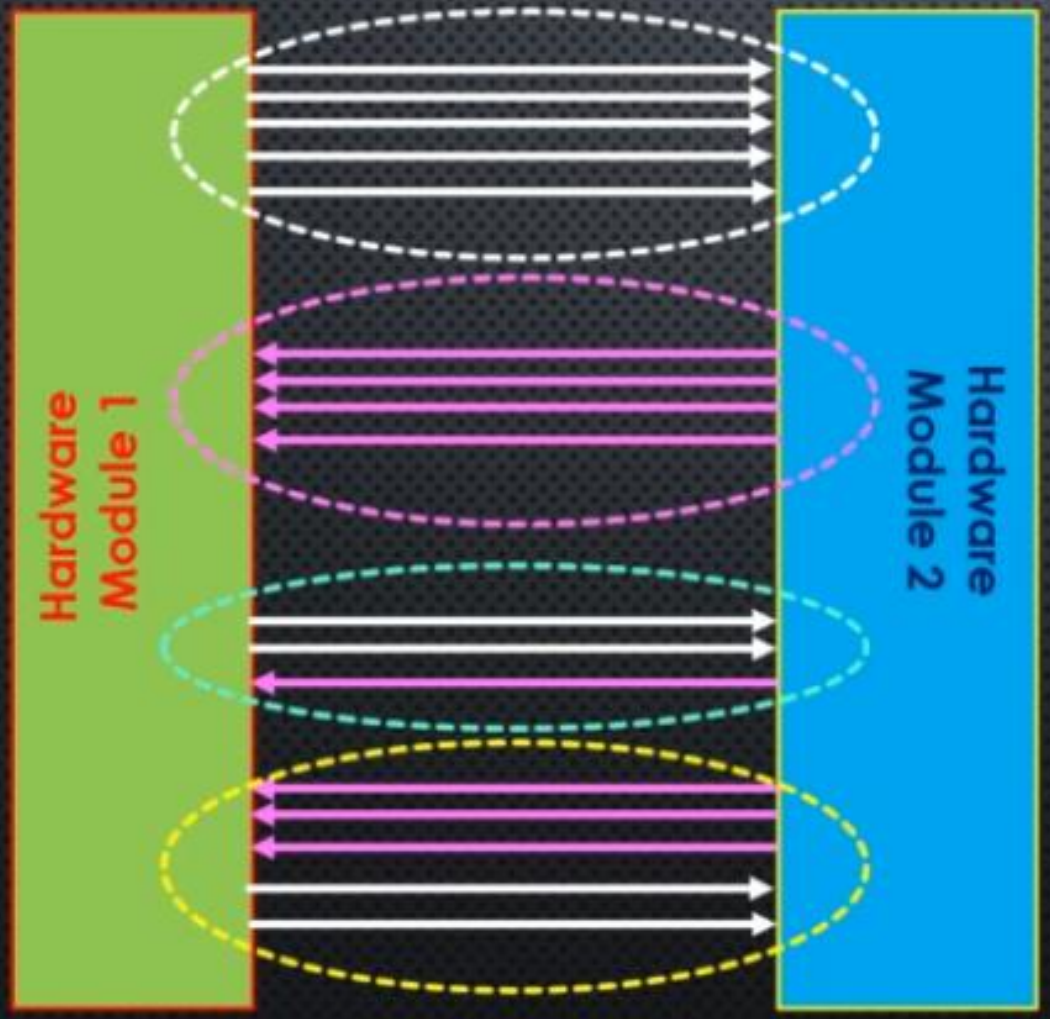
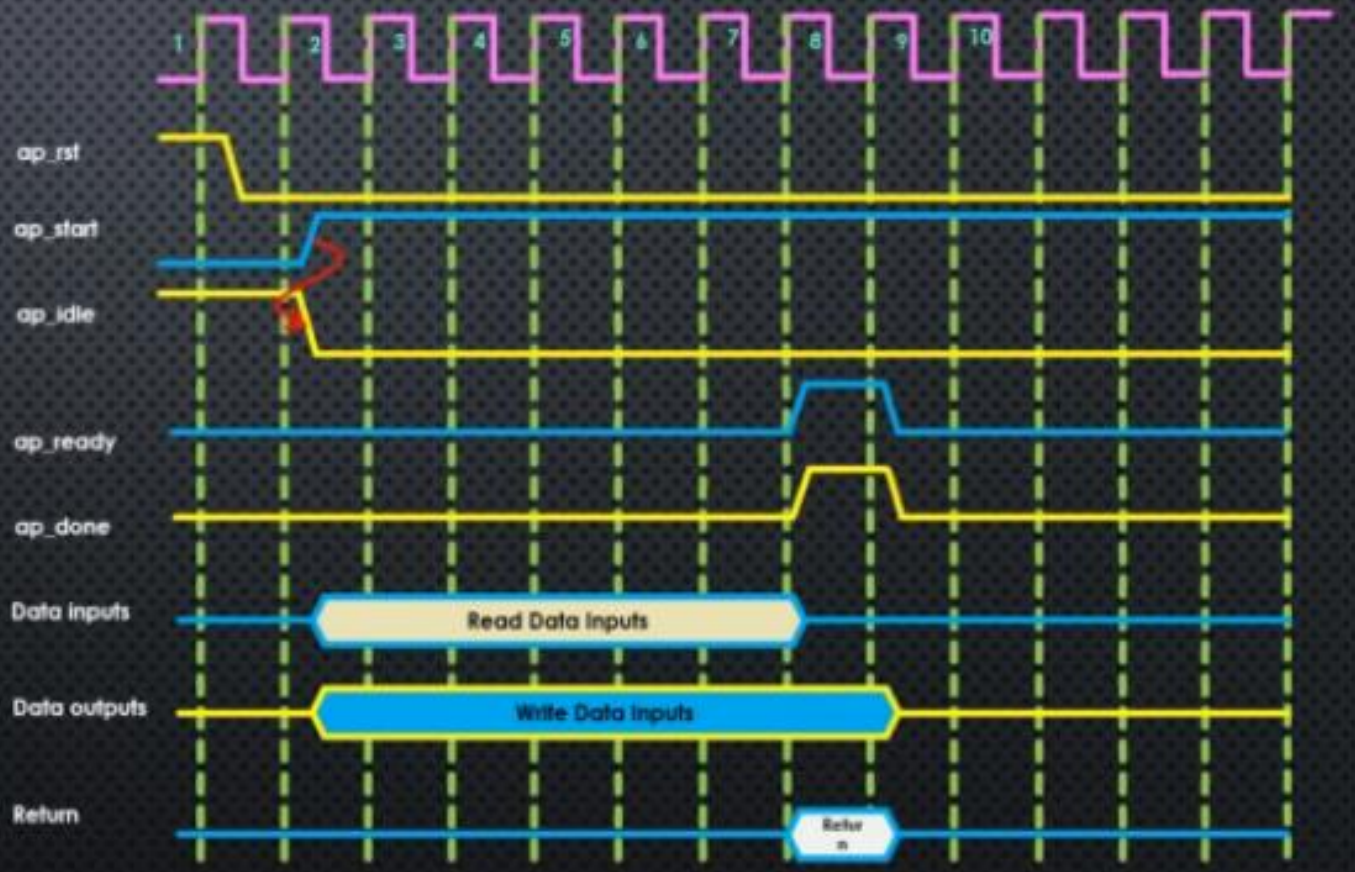
# INTERFACE SYNTHESIS

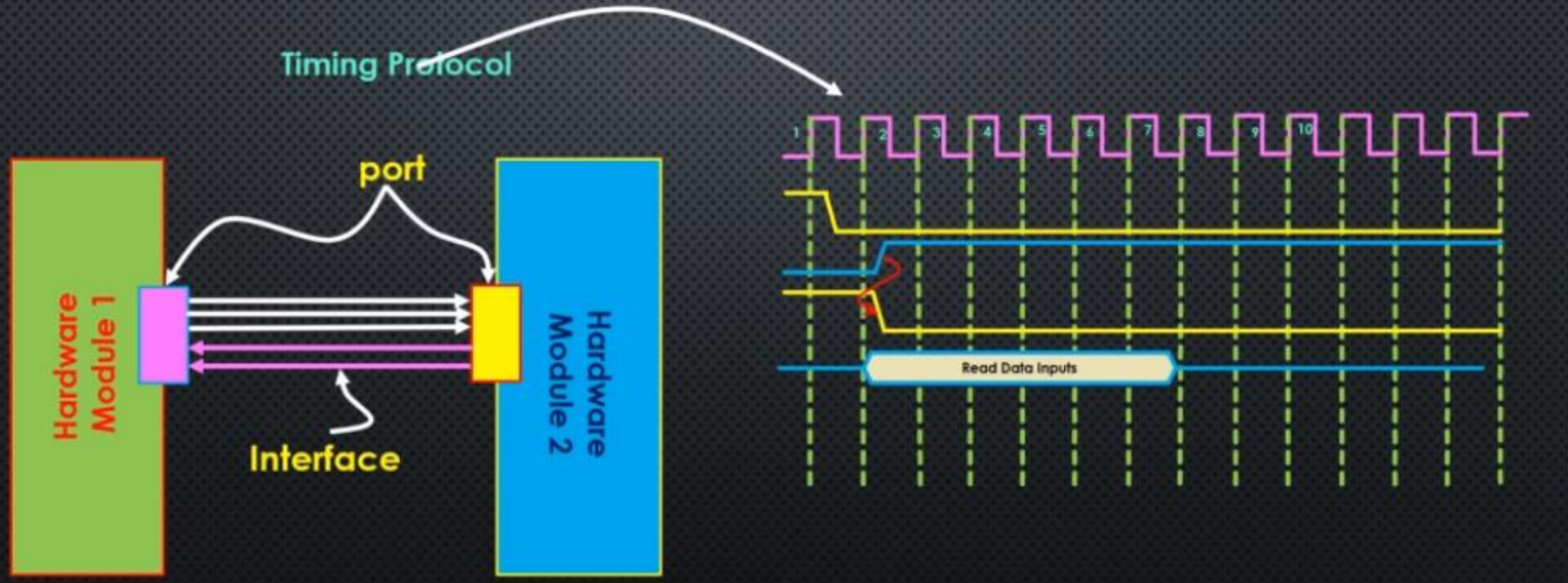
Lecture-15

# HARDWARE MODULE COMMUNICATION

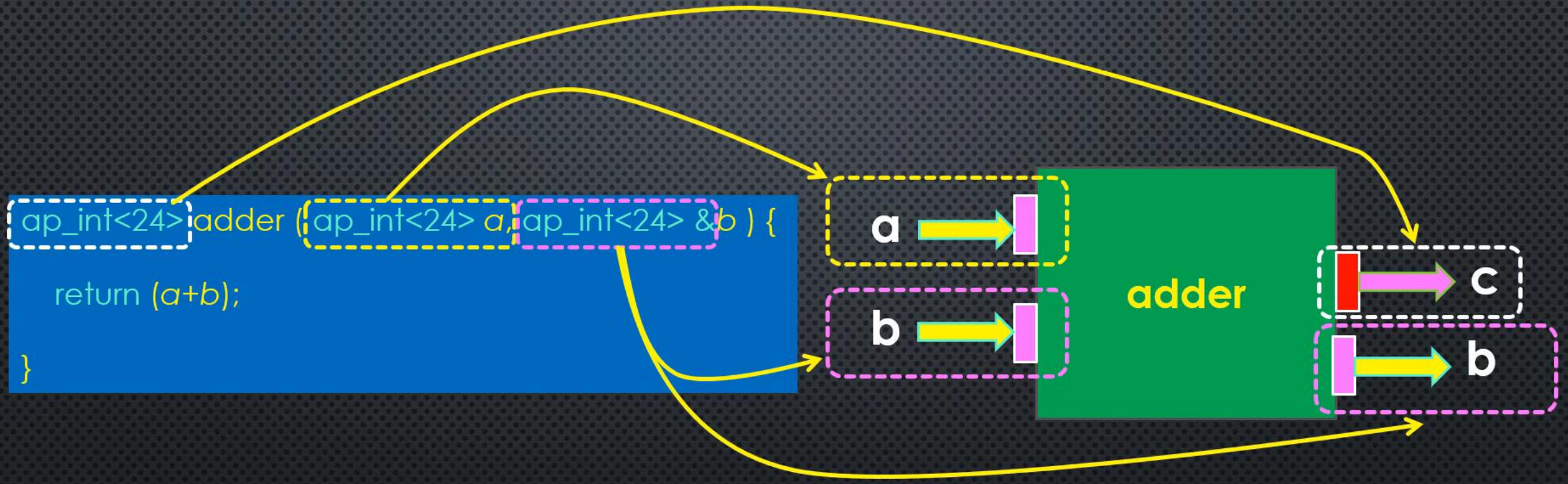


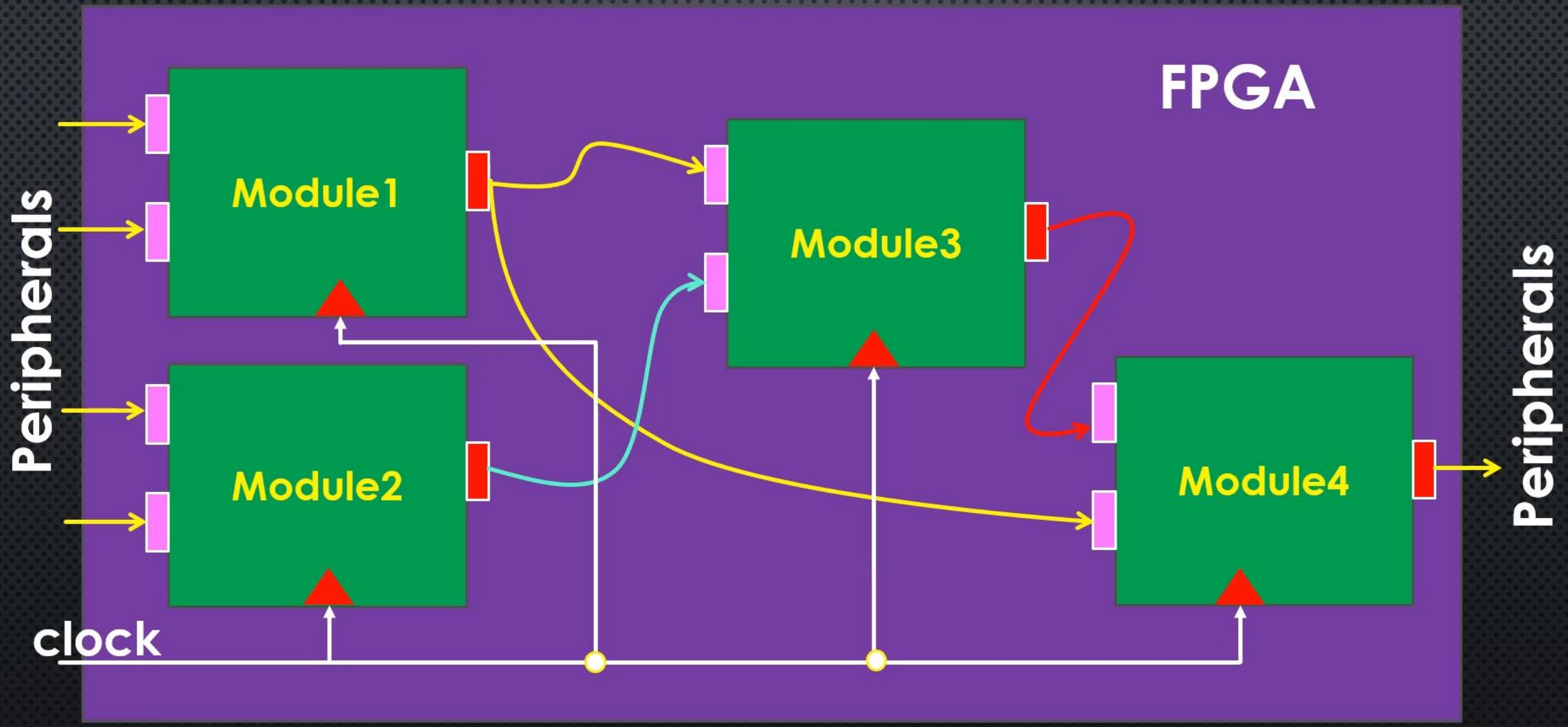
# HARDWARE MODULE COMMUNICATION



# HLS AND HARDWARE MODULE

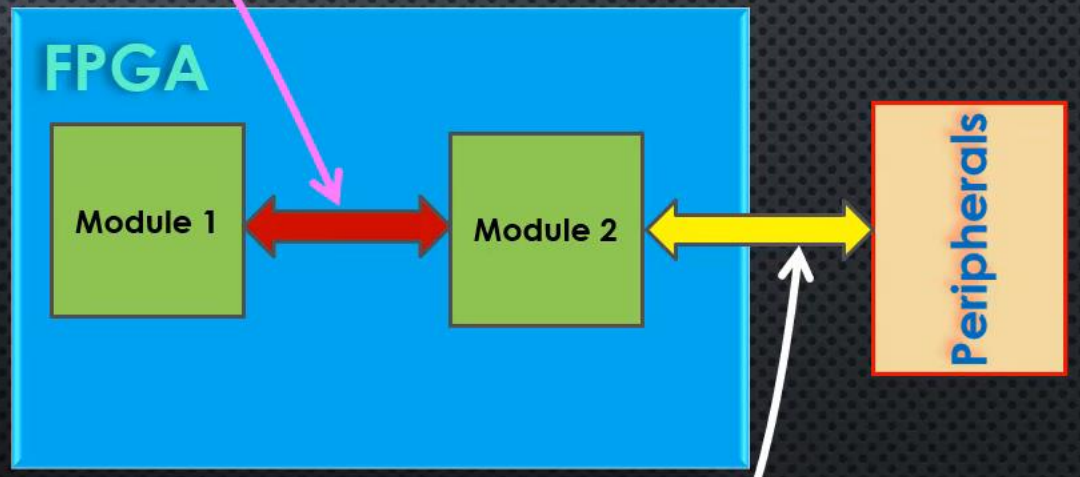


# PERIPHERAL COMMUNICATION



# COMMUNICATIONS

- Communication among modules
- Communication with peripherals



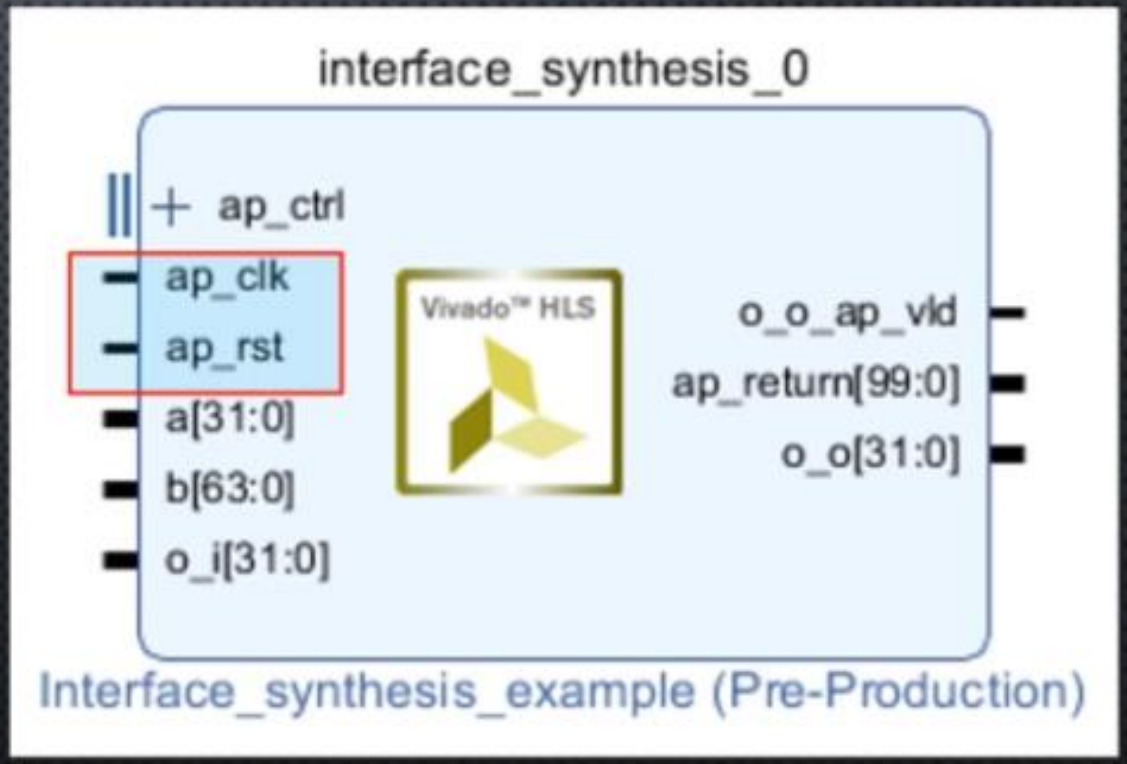
# PORT TYPES

Vitis HLS adds three types of ports to HLS design after synthesis:

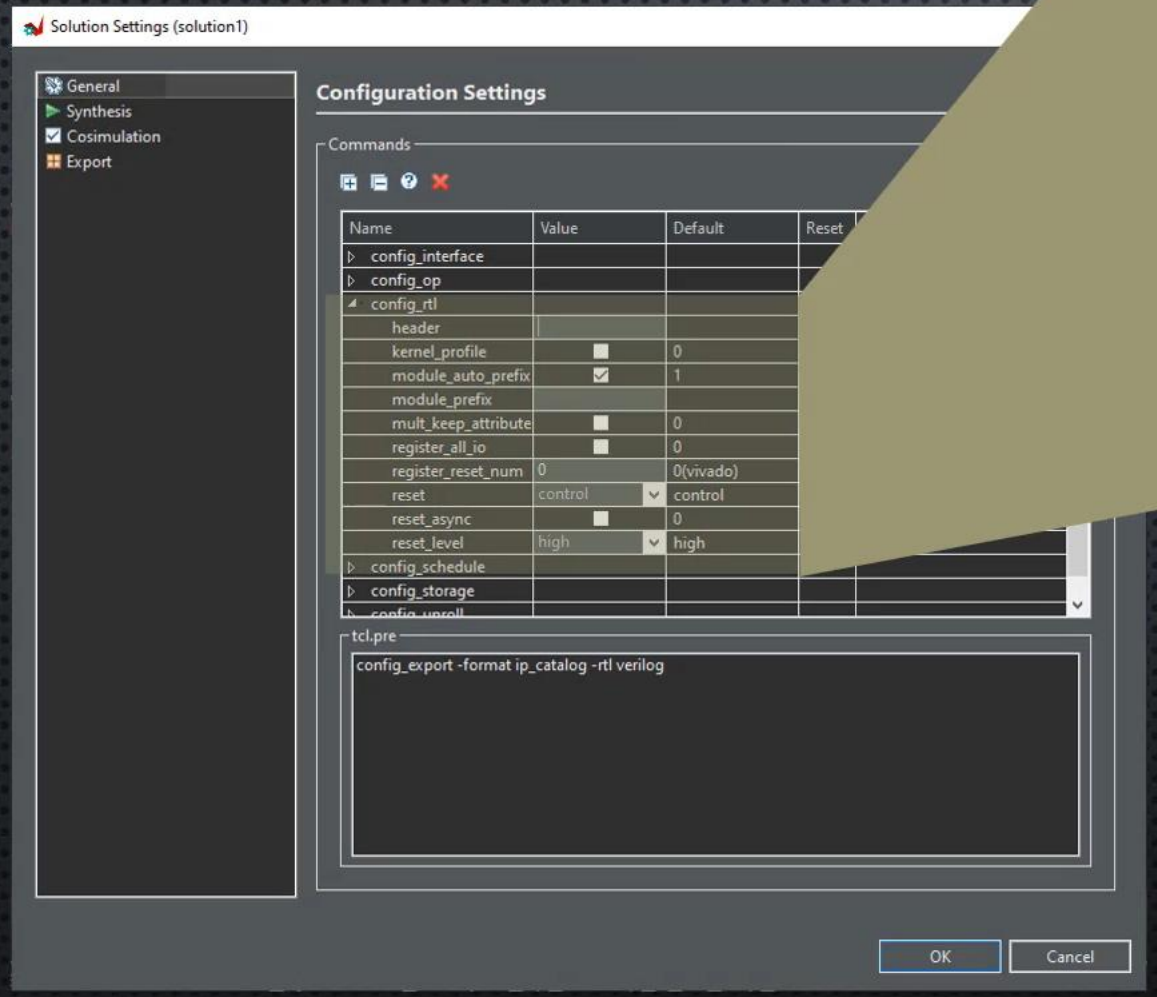
- Clock and Reset ports
- Block-Level ports
- Argument-Level ports

# CLOCK AND RESET PORTS

```
#include <ap_int.h>


ap_int<100> interface_synthesis_example(
    int a,
    long long int b,
    float *o) {

    ap_int<100> r;

    r  = a * b;
    *o = *o / a;

    return r;
}
```

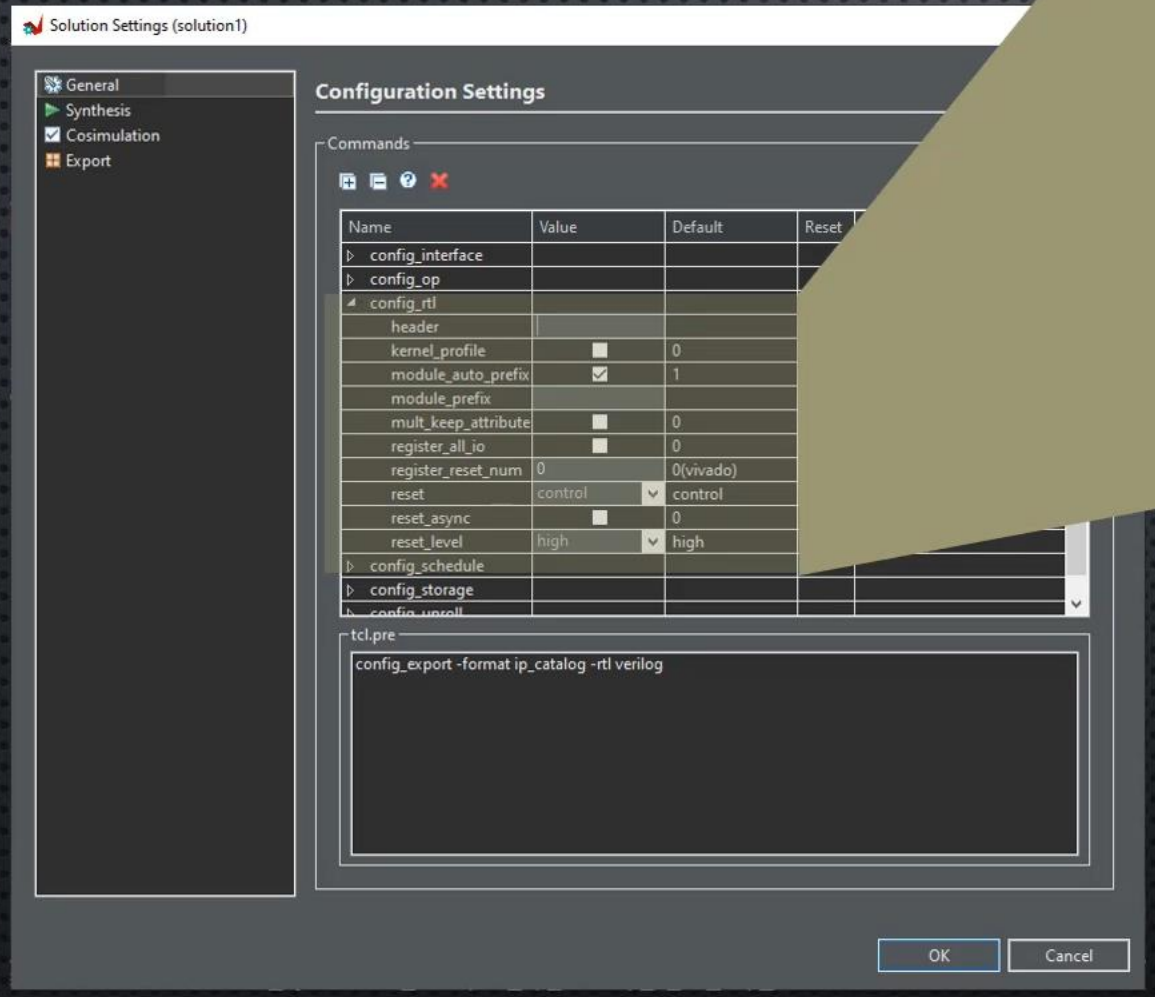


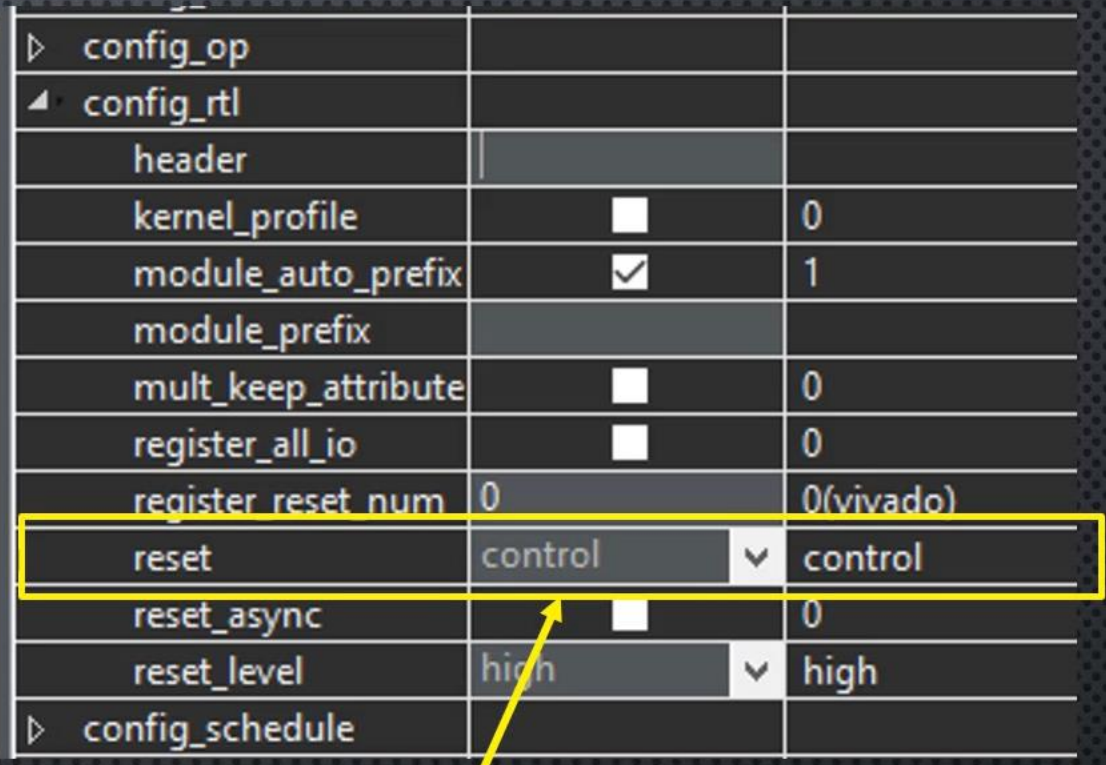
# RESET SETTING




| | | |
|---|---|---|
| config_op | | |
| config_rtl | | |
| header | | |
| kernel_profile | ☐ | 0 |
| module_auto_prefix | ☑ | 1 |
| module_prefix | | |
| mult_keep_attribute | ☐ | 0 |
| register_all_io | ☐ | 0 |
| register_reset_num | 0 | 0(vivado) |
| reset | control | control |
| reset_async | ☐ | 0 |
| reset_level | high | high |
| config_schedule | | |

# RESET SETTING



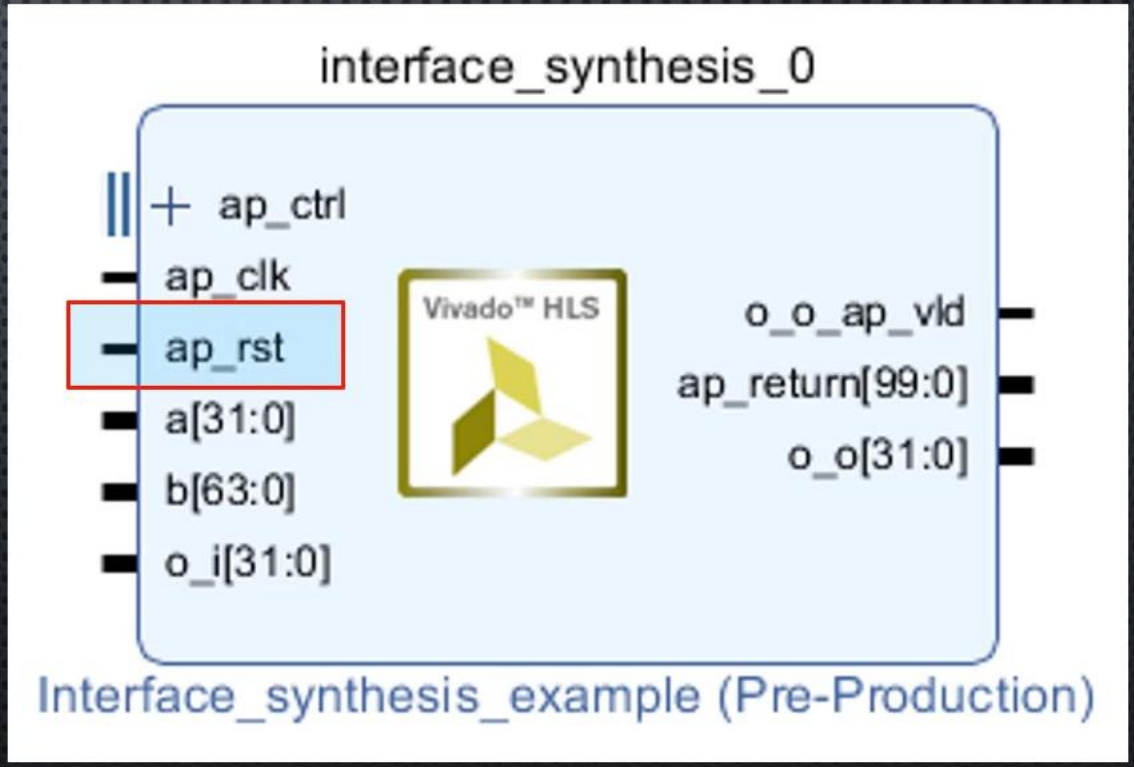




- none
- control
- state
- all

# RESET CONFIGURATION

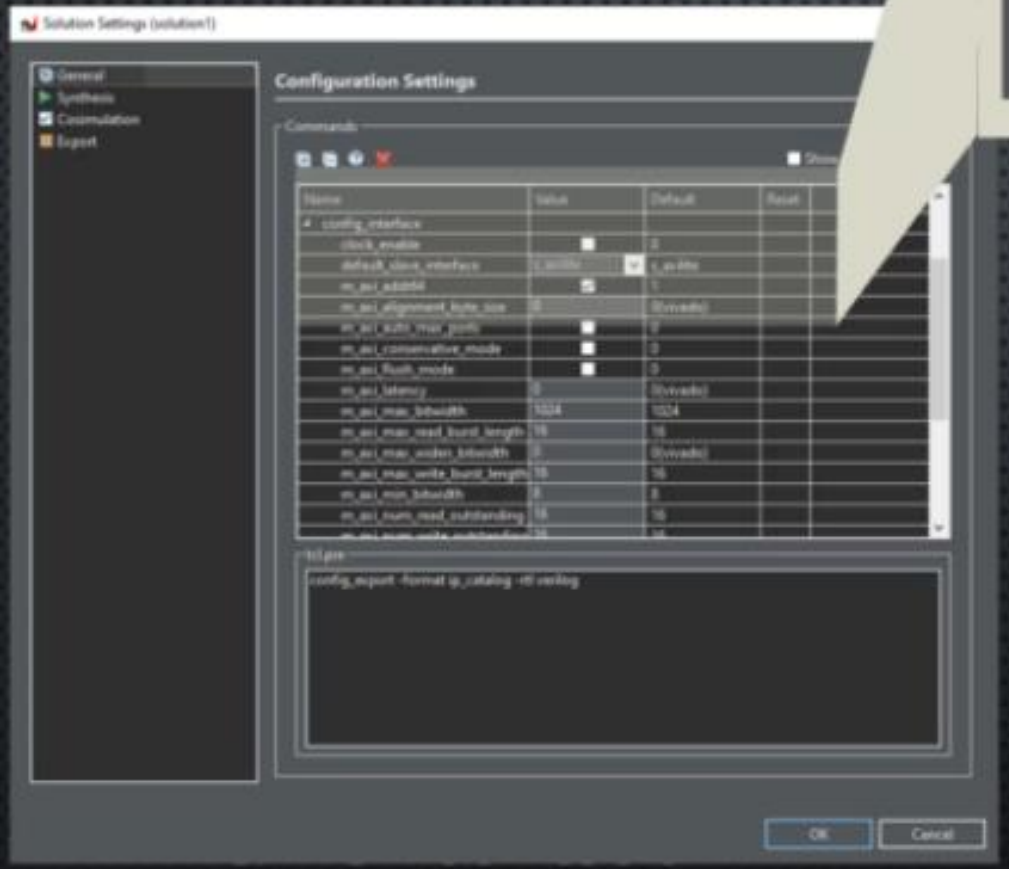
#pragma HLS reset variable=<a> off

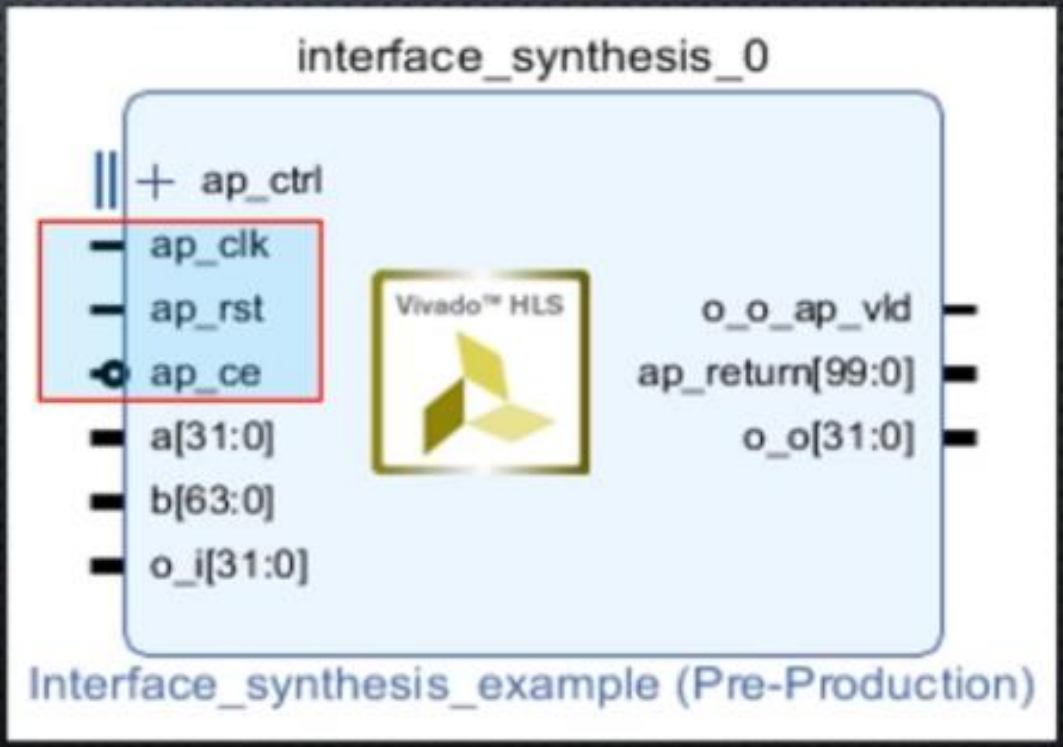
```
#include <ap_int.h>


ap_int<100> interface_synthesis_example(
    int a,
    long long int b,
    float *o) {
 #pragma HLS reset variable=r off
    ap_int<100> r;

    r  = a * b;
    *o = *o / a;

    return r;
}
```



# CLOCK ENABLE PORT

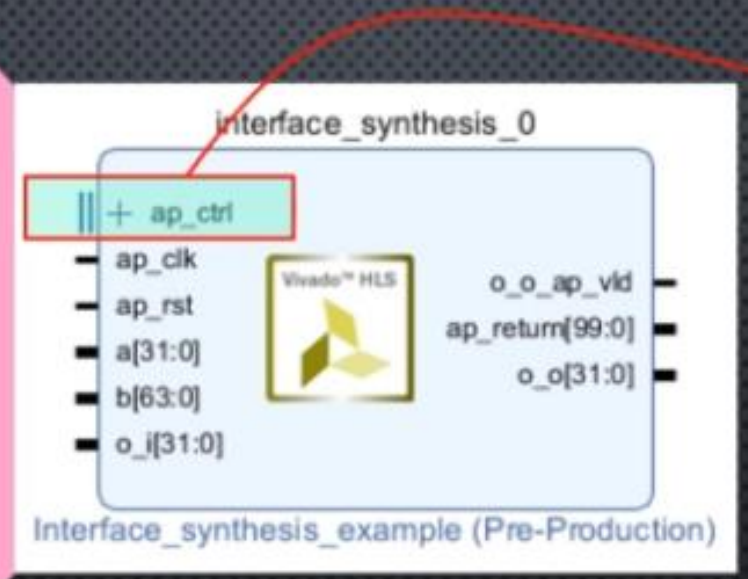
| Name | Value | Default |
|---|---|---|
| config_interface | | |
| clock_enable | ☑ | 0 |
| default_slave_interface | s_axilite | s_axilite |
| m_axi_addr64 | ☑ | 1 |
| m_axi_alignment_byte_size | 0 | 0(vivado) |

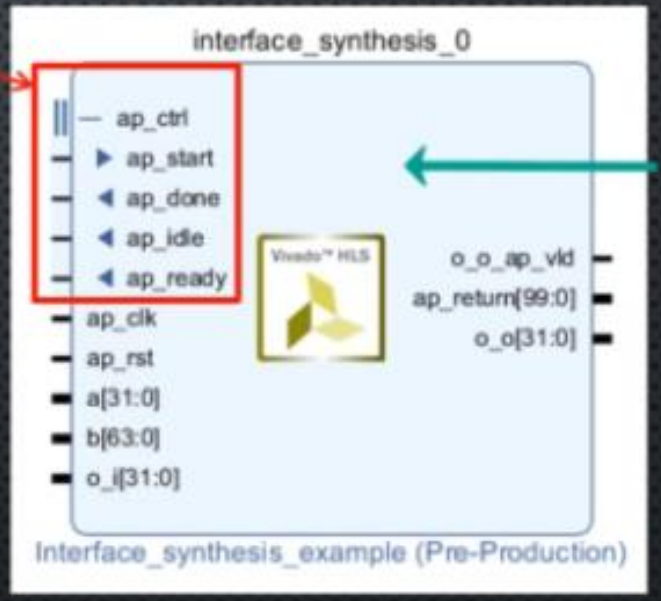


# BLOCK-LEVEL PORT

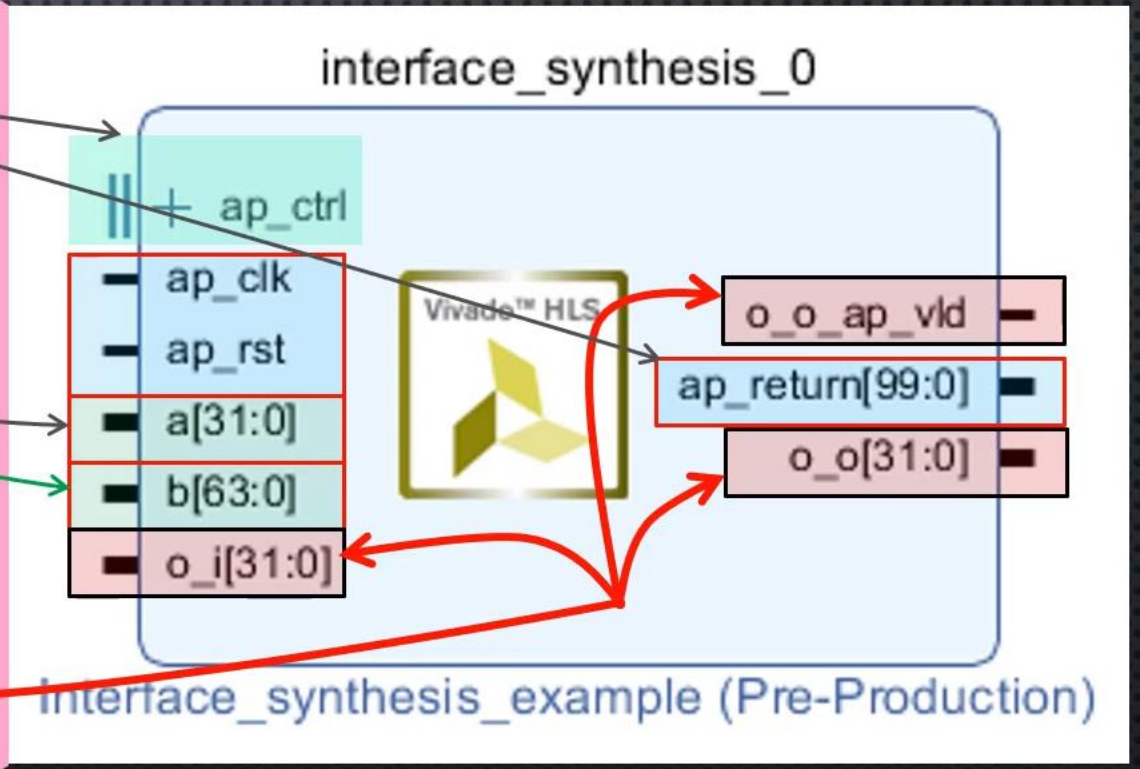
```
#include <ap_int.h>


ap_int<100>
interface_synthesis_example(
    int a,
    long long int b,
    float *o) {

    ap_int<100> r;

    r  = a * b;
    *o = *o / a;

    return r;
}
```



# EXAMPLE (PORTS)

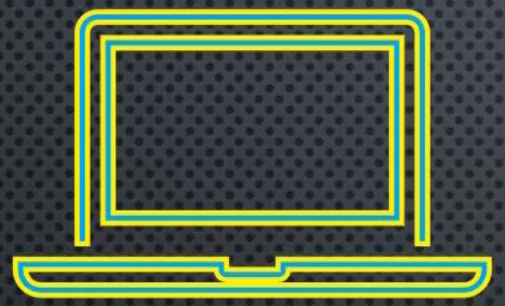
```
#include <ap_int.h>


ap_int<100> interface_synthesis_example(
    int a,
    long long int b,
    float *o) {

    ap_int<100> r;

    r  = a * b;
    *o = *o / a;

    return r;
}
```

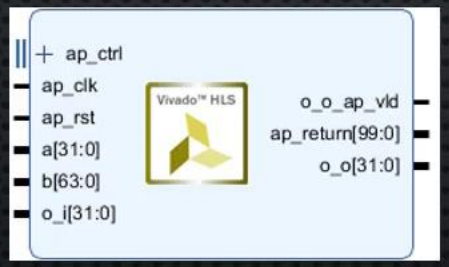


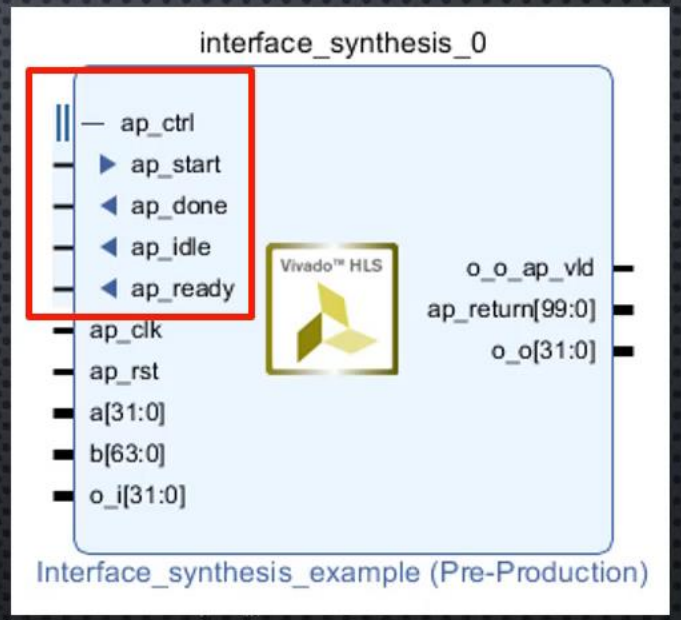
# TAKEAWAY

- The reset signal behaviour can be controlled through solution settings and adding the corresponding pragma in code
- The HLS tool considers a default interface for block and arguments if you don’t define them.

# BLOCK-LEVEL INTERFACE PROTOCOLS





# BLOCK-LEVEL I/O PROTOCOLS

**Block-level interface types:**

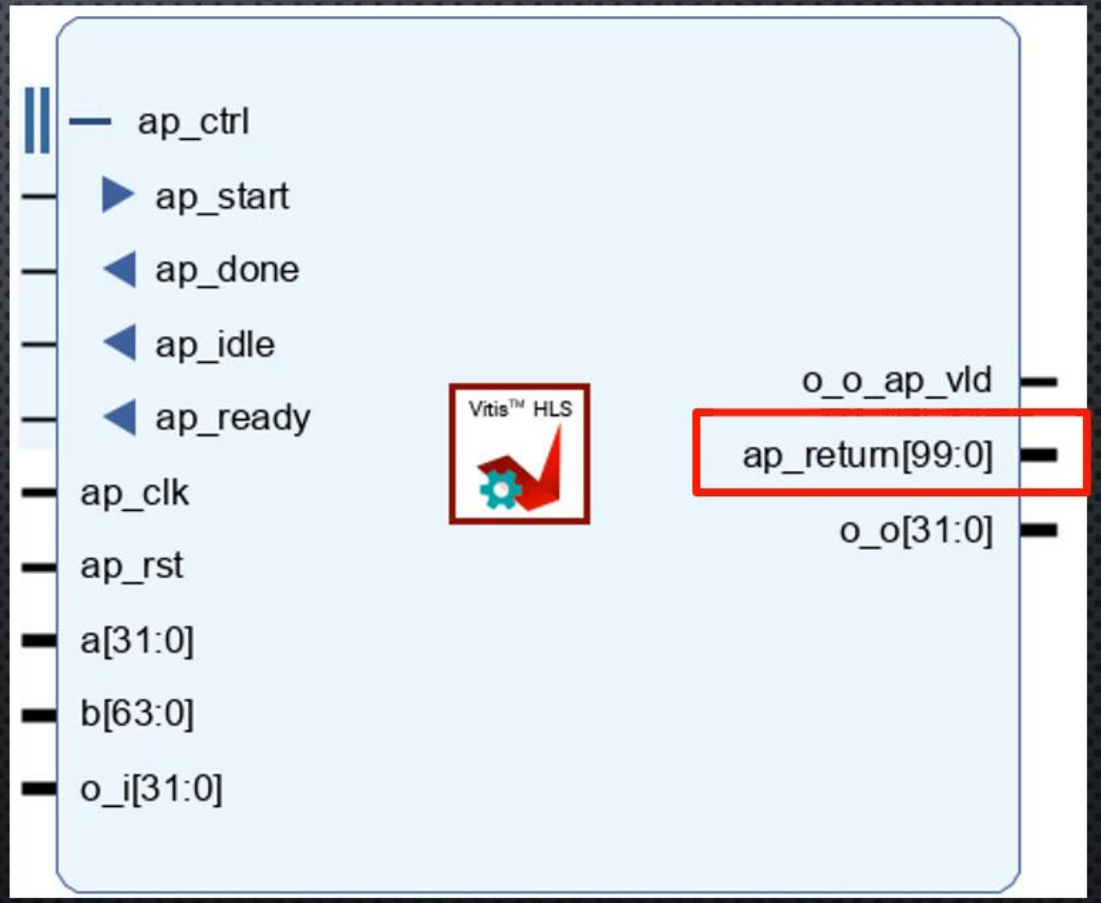
- ap_ctrl_none
- ap_ctrl_hs
- ap_ctrl_chain
- s_axilite

**Block-level handshake signals specify the following:**

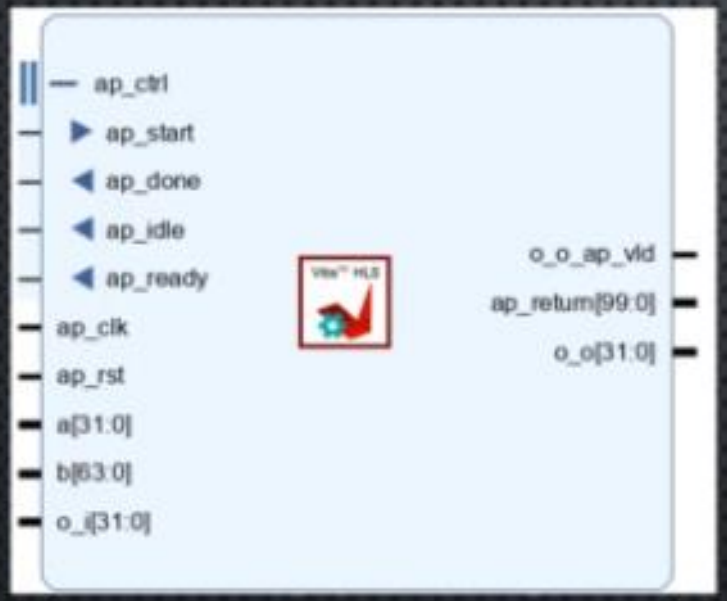
- When the design can start to perform the operation
- When the operation ends
- When the design is idle and ready for new inputs

# BLOCK-LEVEL I/O PROTOCOLS

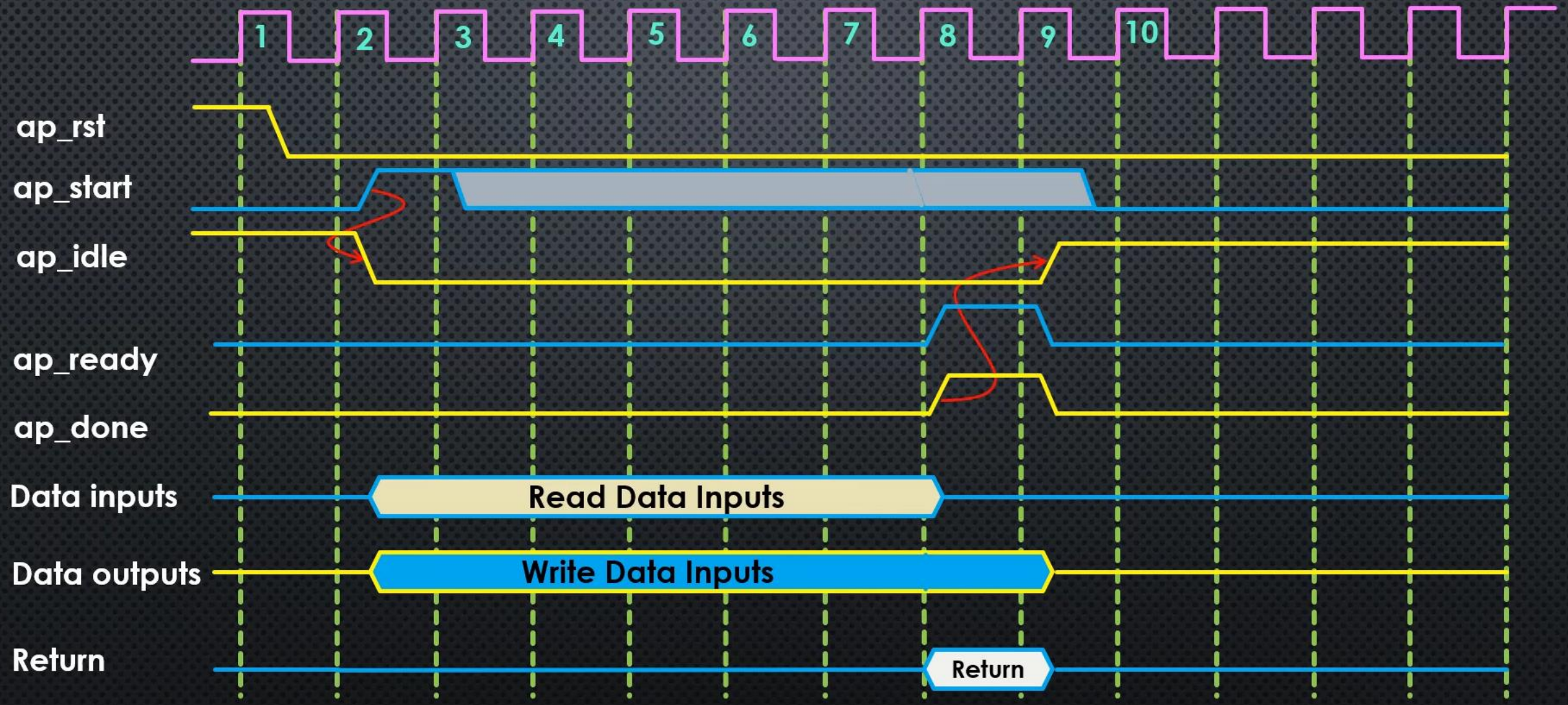
```
#include <ap_int.h>


ap_int<100> interface_synthesis_example(
    int a,
    long long int b,
    float *o) {
#pragma HLS INTERFACE ap_ctrl_hs port=return
    ap_int<100> r;

    r  = a * b;
    *o = *o / a;

    return r;
}
```



# BLOK-LEVEL AP_CTRL_HS INTERFACE

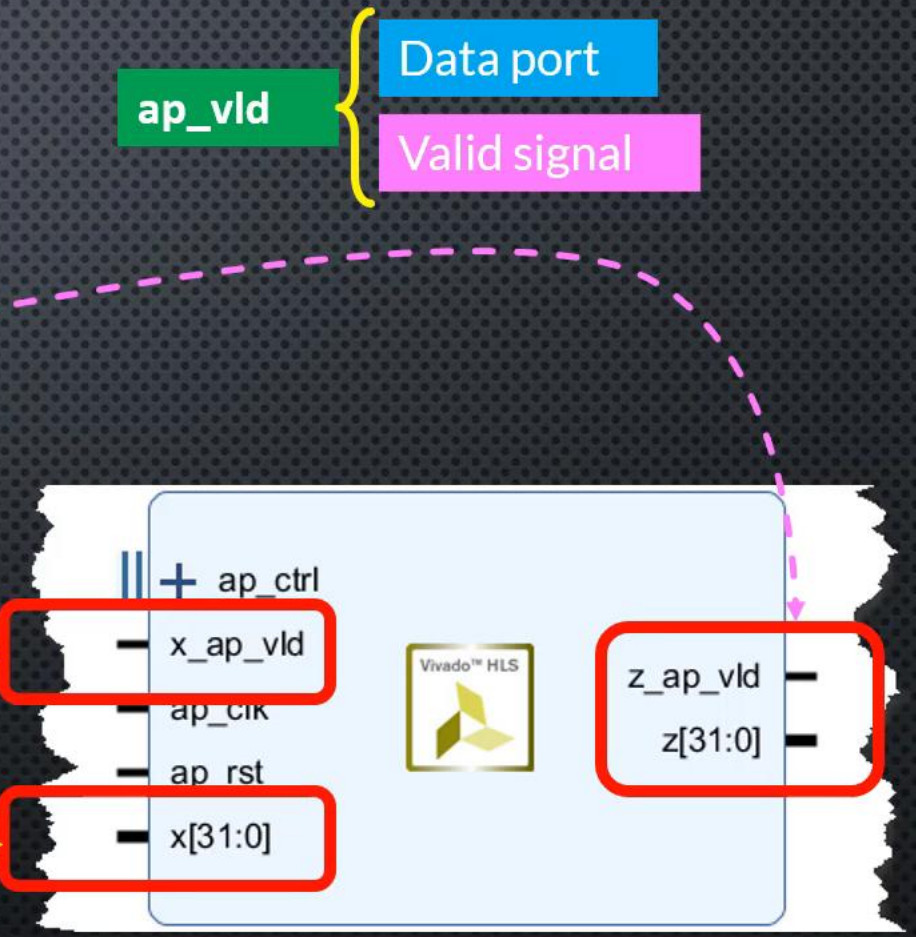



# BLOK-LEVEL AP_CTRL_HS SIGNALLING

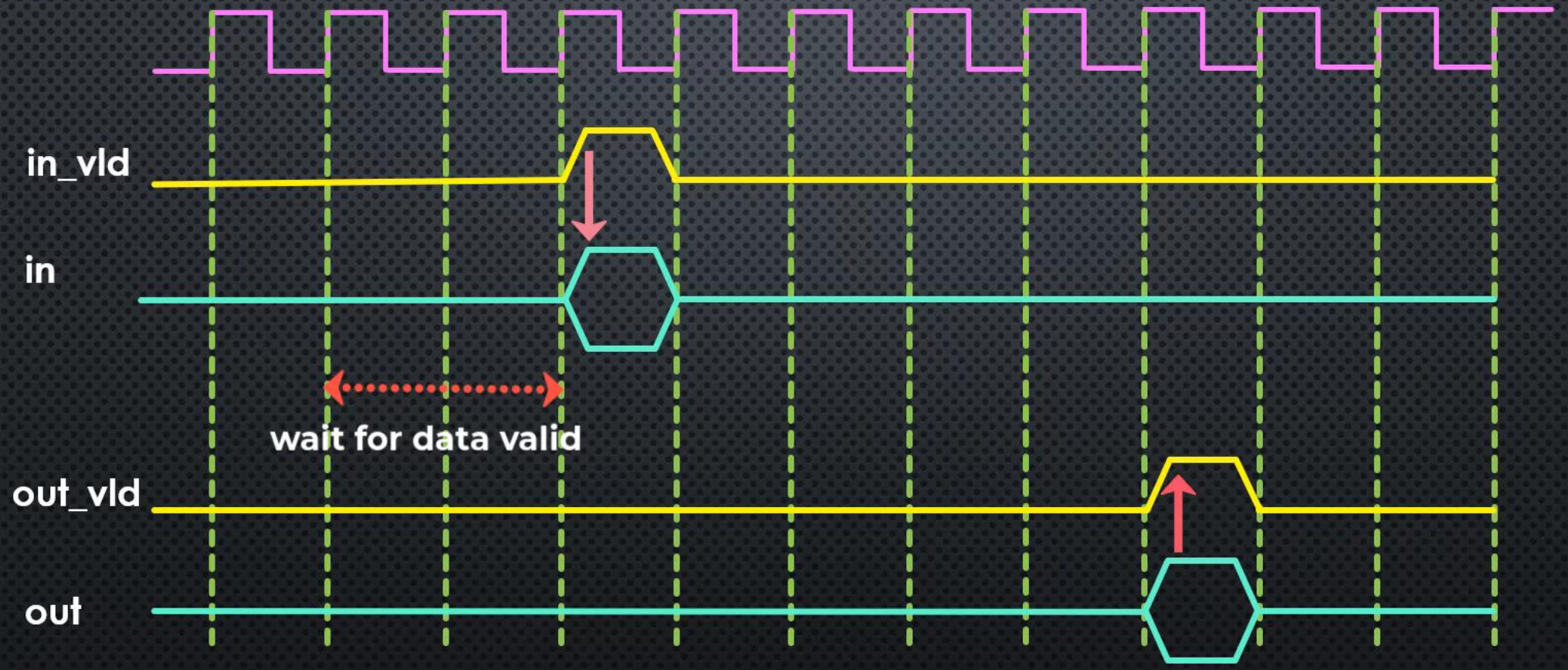


# PORT LEVEL (AP_VLD)

```
void ap_vld_example(float x, float &z) {
#pragma HLS INTERFACE ap_ctrl_hs port=return

#pragma HLS INTERFACE ap_vld port=x
#pragma HLS INTERFACE ap_vld port=z


  z = x + 4;
}
```



# PORT LEVEL (AP_VLD) TIMING



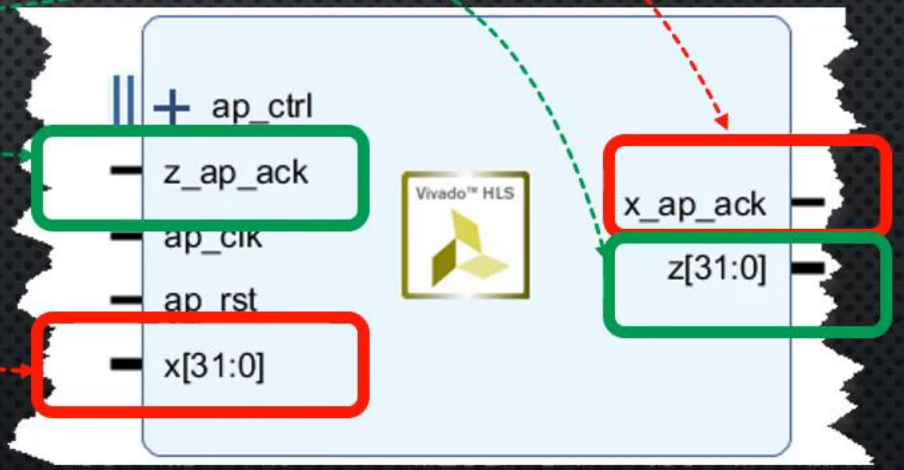
# PORT LEVEL (AP_ACK)

```
void ap_vld_example(float x, float &z) {
#pragma HLS INTERFACE ap_ctrl_hs port=return

#pragma HLS INTERFACE ap_ack port=x
#pragma HLS INTERFACE ap_ack port=z


  z = x + 4;
}
```

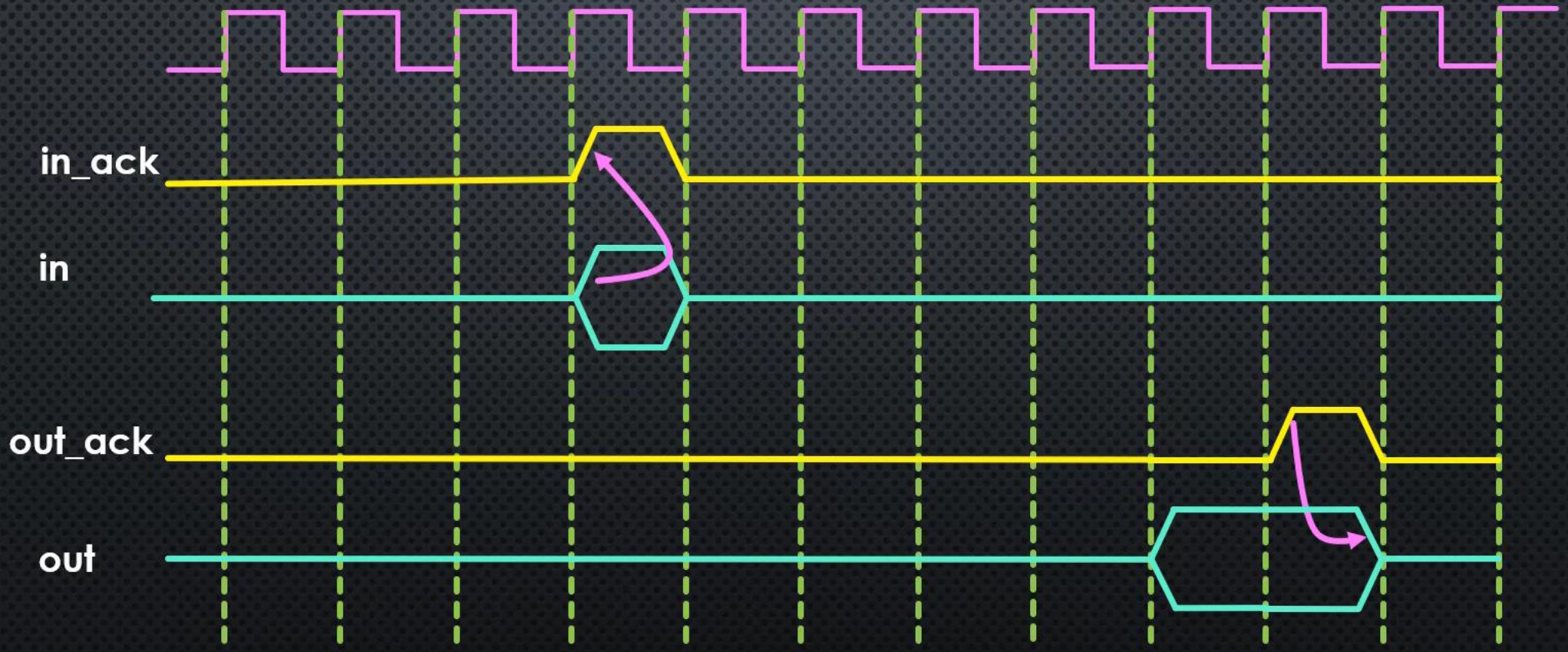
ap_vld

Data port

acknowledge signal



# PORT LEVEL (AP_ACK) TIMING

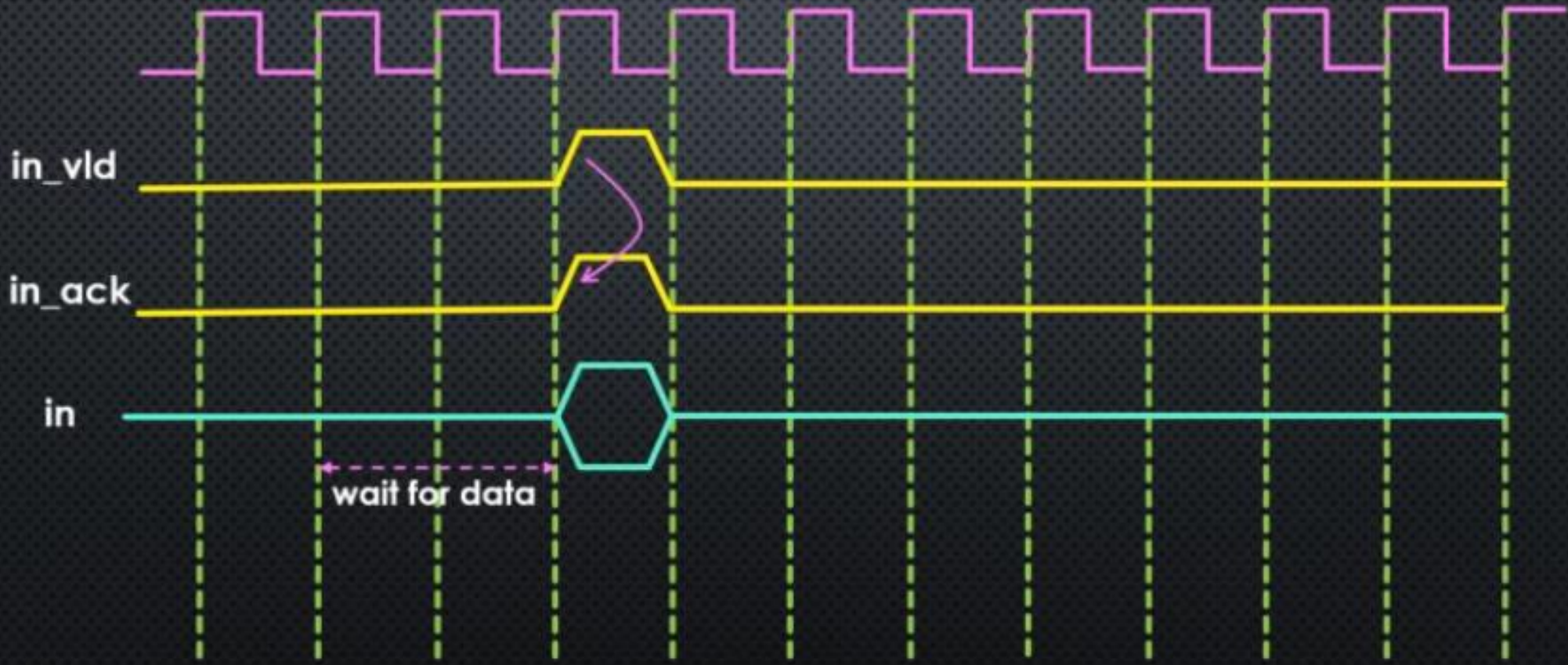
# PORT LEVEL (AP_ACK)

For input arguments, the design generates an output acknowledge that is active-High in the cycle the input is read.

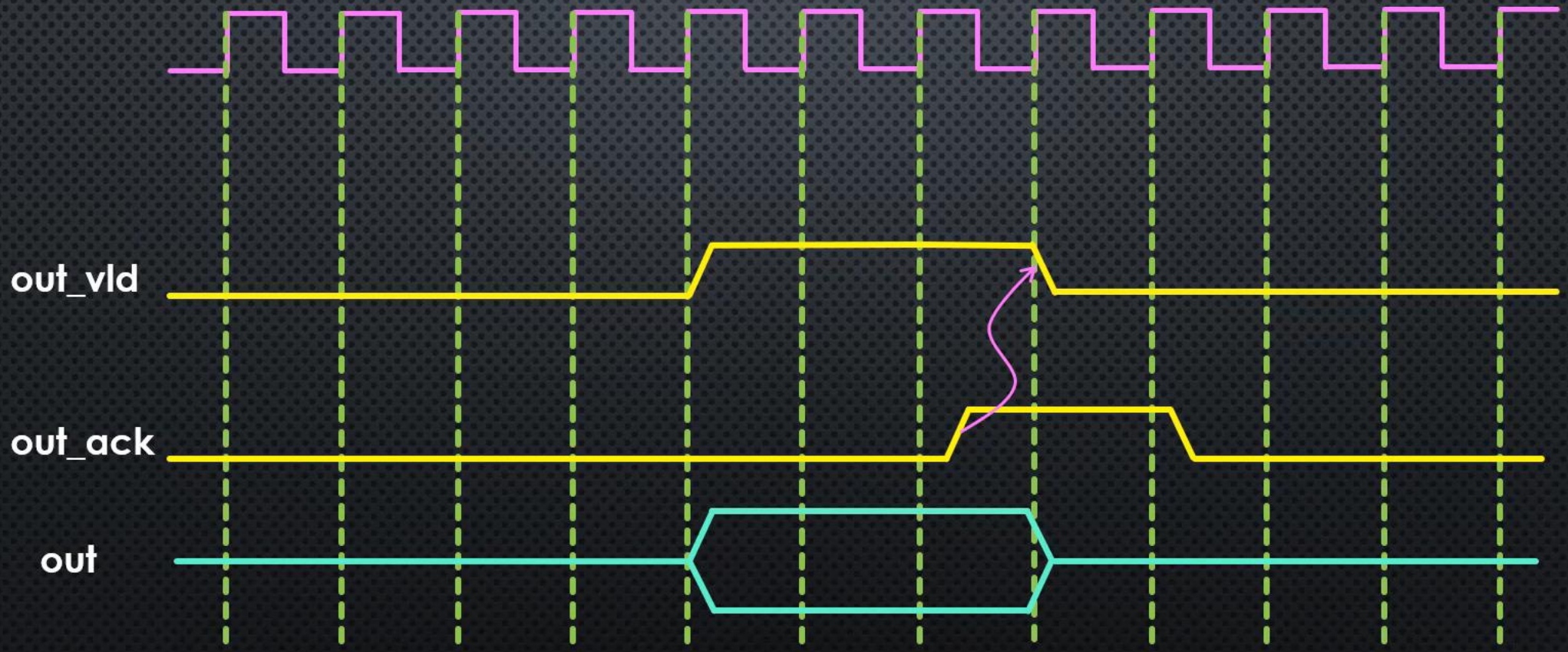
For output arguments, Vivado HLS implements an input acknowledge port to confirm the output was read.

# PORT LEVEL (AP_HS)

# INPUT PORT LEVEL (AP_HS) TIMING

# OUTPUT PORT LEVEL (AP_HS) TIMING

# TAKEAWAY

- Interface mode ap_hs includes a two-way handshake signal with the data port. The handshake is an industry-standard valid and acknowledges handshake.

- The ap_hs port-level I/O protocol provides the following signals:
  - Data port
  - Acknowledge signal to indicate when data is consumed.
  - Valid signal to indicate when data is read.

# Any Question...

# Thank you