MOALU CENTRAL UNIT

by Gatti Marco

Goals

Introduction

MOALU consists of a central unit that performs the following functions on two numbers *A* and *B* of *Nb* bits:

- Sum (*A+B*)
- Complement to 2 (*C2*) of the number A (or of the number B, as desired)
- Subtraction (*A-B*, or *B-A*), after converting the second operand into a negative integer in *C2*
- Comparison (*out=1* if *A=B*)

Goals

010	sum
011	c2
100	subtraction
101	comparison

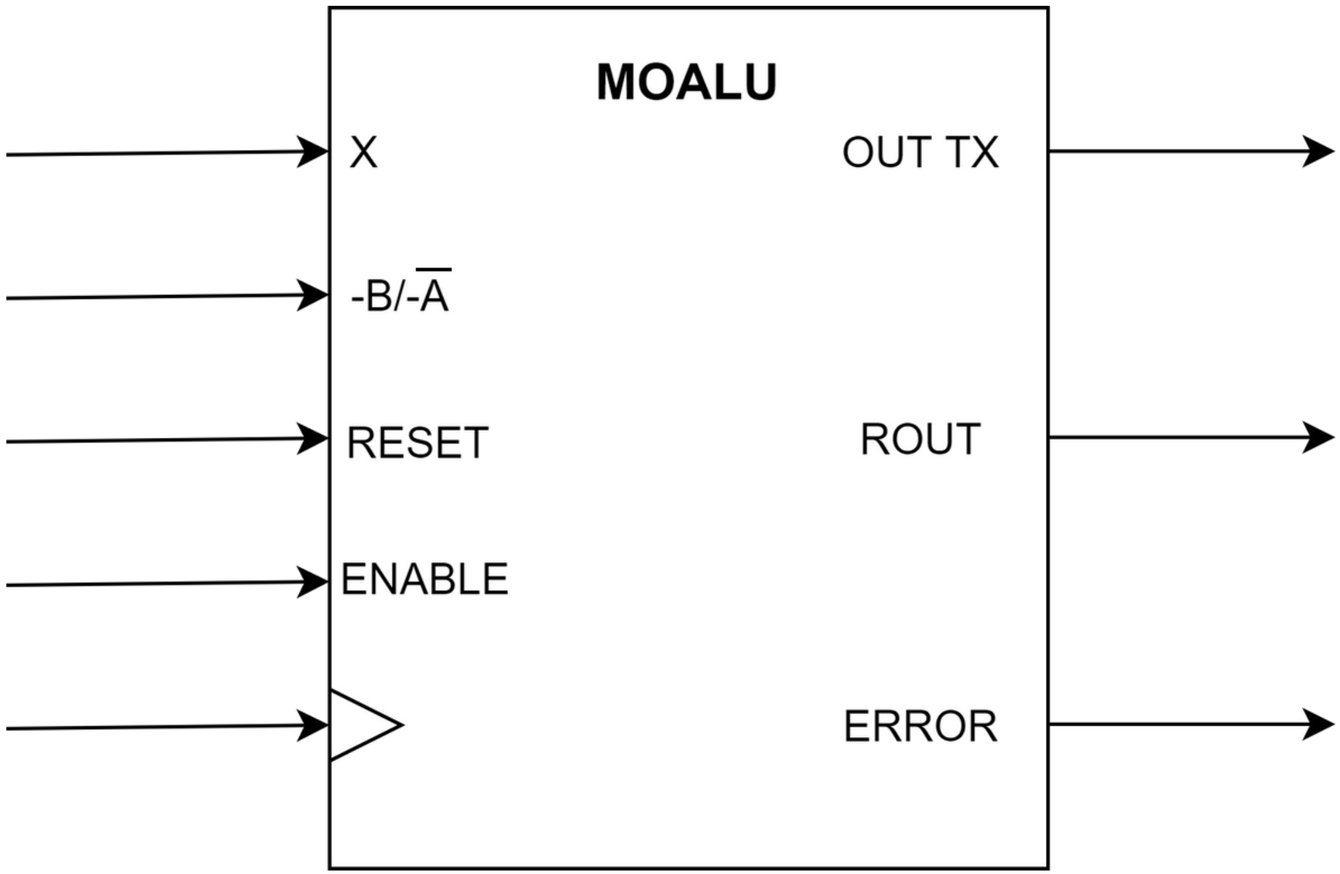
FSM

MOALU uses a serial input interface X and through a Finite-State-Machine (FSM) proceeds as follows:

- In the presence of 3 consecutive zeros, the system is in stand-by, does not change the state of the registers and the outputs of MOALU are in hold
- If there are 3 consecutive ones, the system is in receiver mode (RX) and saves A and B numbers.
- In the presence of sequence 001, the system is in transmitter mode (TX), and outputs A and B in serial mode
- In the presence of other ternary sequences, the system operates as in the table above, saving the result of the operation in an output register ROUT

Moreover, MOALU is equipped with a global, asynchronous reset signal that resets the contents of the RA-RB registers and returns the system to Stand-By mode.

MOALU symbol



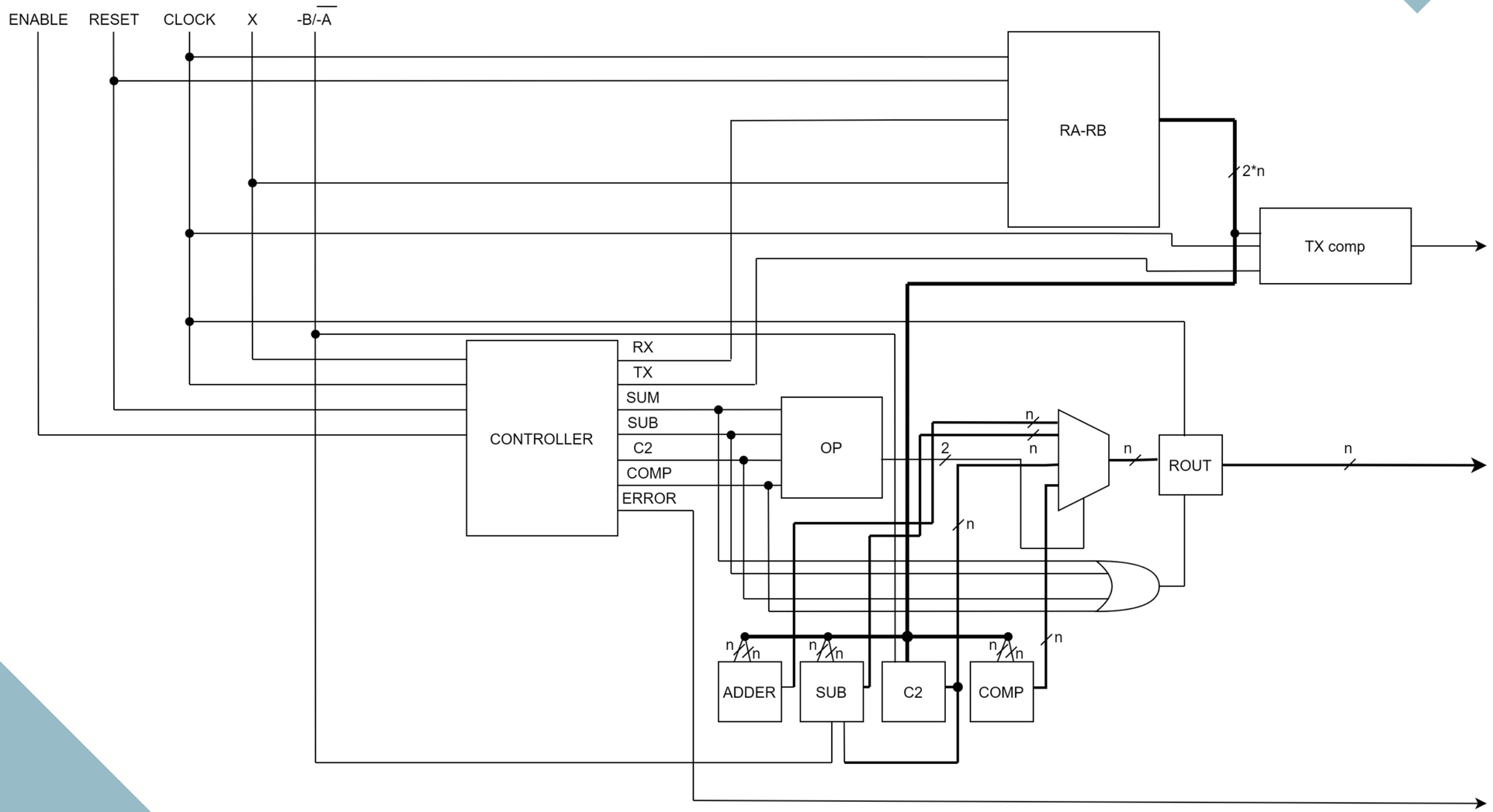
INPUTS

Signal name	Bit resolution	Description
X	1	Serial interface used to enter operation codes, or, when MOALU operates in RX mode, the numbers *A* and *B* .
-B/-$\overline{A}$	1	Selector to decide whether to apply the two's complement to number *A* or number *B*
RESET	1	Global asynchronous reset, which resets the *RA* and *RB* registers to zero and restores the system to Stand-By mode
ENABLE	1	Simple enable signal, which enables **MOALU** to operate
CLOCK	1	Periodic signal of a given frequency, to enable **MOALU** to operate synchronously

OUTPUTS

Signal name	Bit resolution	Description
OUT TX	1	Provides serial output of the stored *A* and *B* numbers, when MOALU is in TX mode
ROUT	1	Outputs the value stored in the *ROUT* register
ERROR	1	*S*ingle bit to indicate that **MOALU** is in an error state

block diagram

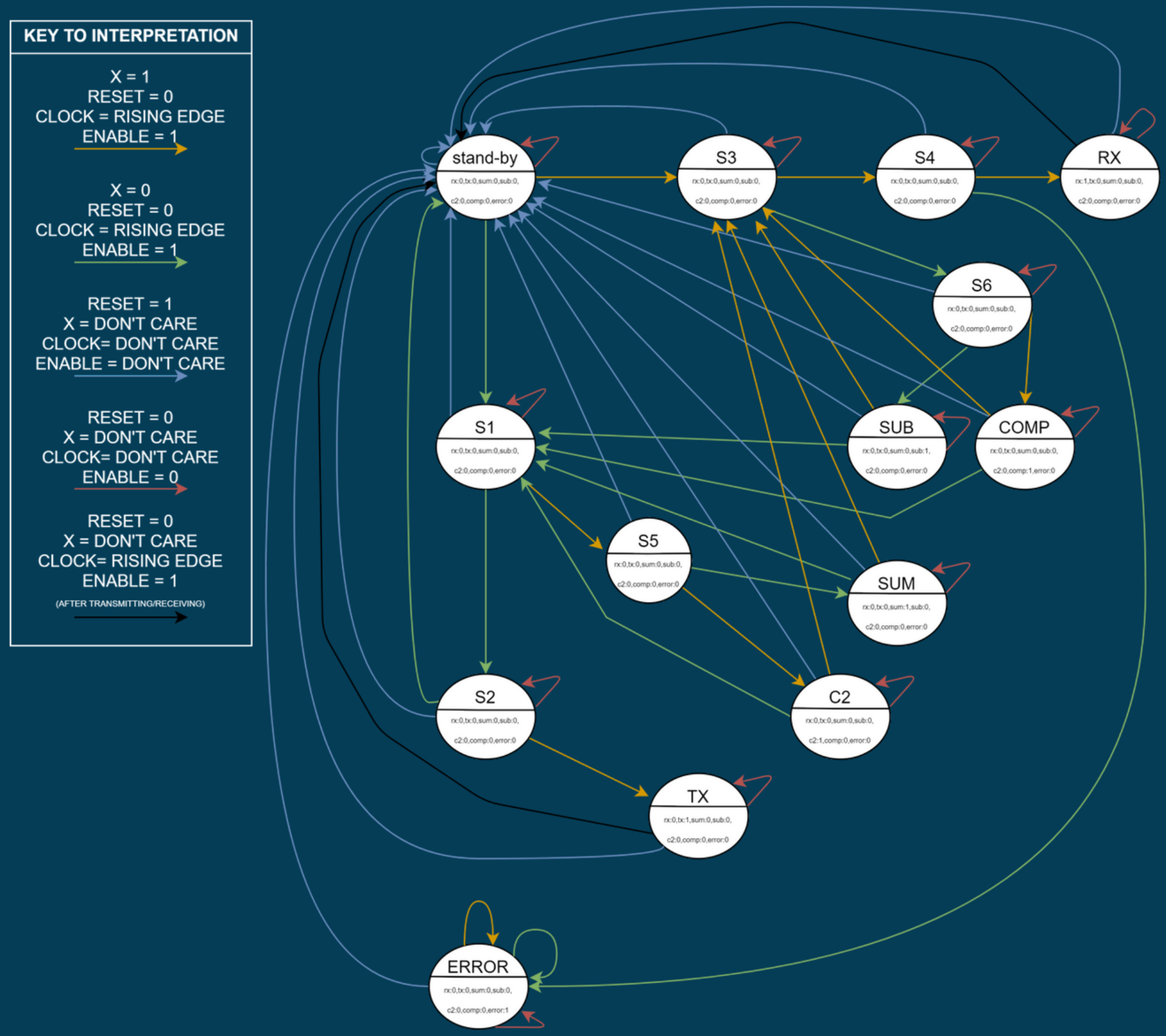


COMPONENTS

Component	Input	Output	Description
CONTROLLER (FSM)	Enable, Clock, Reset, X	RX, TX, SUM, SUB C2, COMP, ERROR (1 bit resolution for all)	If *enable* is high, it synchronously interprets the serial input and activates the desired state. With asynchronous reset it activates the *Stand-By* state
OP	SUM, SUB C2, COMP	1-bit selector <0:1>	Select which operation is to be stored in *ROUT* register
ROUT (PIPO)	DATA IN <0:NB-1> Clock, Rout-Enable	ROUT <0:NB-1>	Register which stores the result of the operation performed
ADDER	A<0:NB-1>, B<0:NB-1>	Result <0:NB-1>	Performs the sum
SUB	A<0:NB-1>, B<0:NB-1>, C2 <0:NB-1>, -A/-$\overline{B}$	Result <0:NB-1>	Performs the subtraction
C2	AB<0:2*NB-1>, -A/-$\overline{B}$	Result <0:NB-1>	Performs the two's complement
COMP	A<0:NB-1>, B<0:NB-1>	Result <0:NB-1>	Performs the comparison
RA-RB (SIPO)	Clock, Reset, X, RX	AB<0:2*NB-1>	If *RX* status is on it stores the A and B values in a serial way and returns them in parallel. Asynchronous reset sets the register to zero
TX COMP	AB<0:2*NB-1>, Clock, TX	OUT TX	When in TX state, it gives back the A and B values in serial mode

WE USE THE FINITE STATE DIAGRAM TO DESIGN THE CONTROLLER

FINITE STATE DIAGRAM



4-bit per number

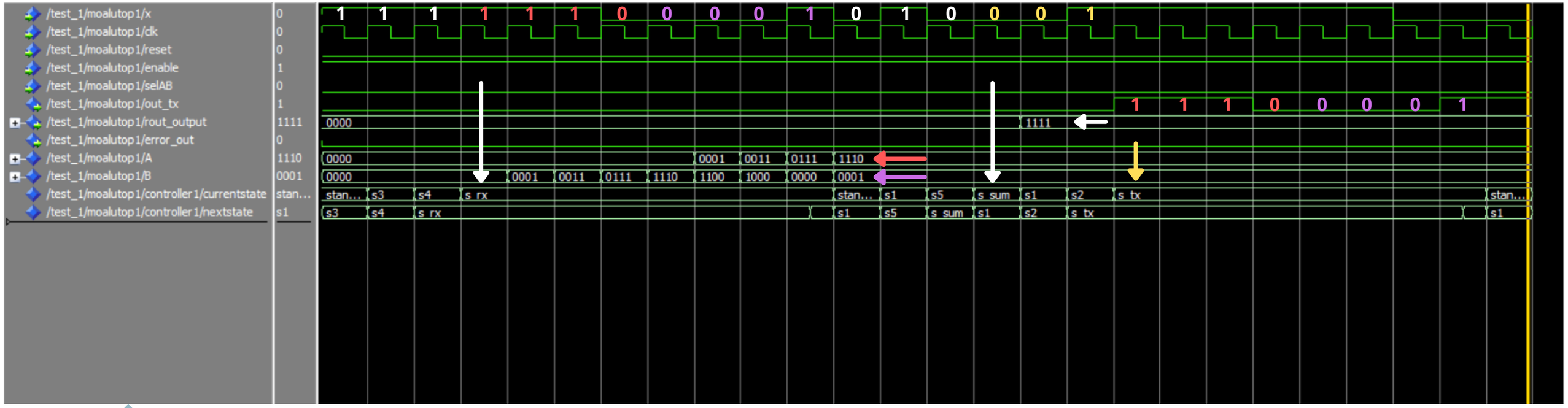
The generic parameter, is now fixed to 4

A and B

For simplicity's sake, the first two tests have the value of A: 1110 and the value of B: 0001, while in the last test both have the value: 1110.

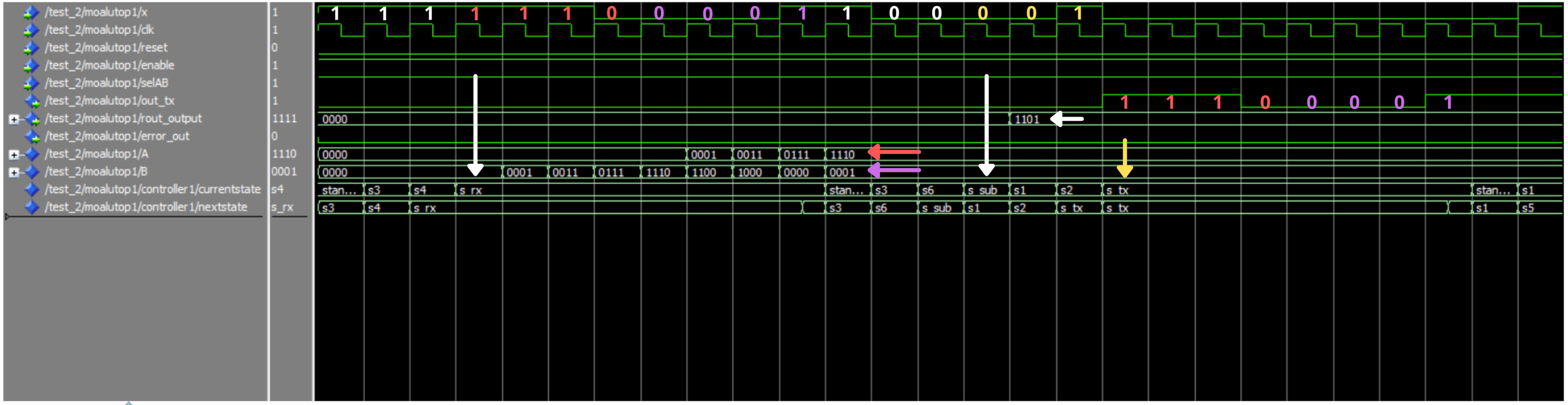
TESTS

TEST 1



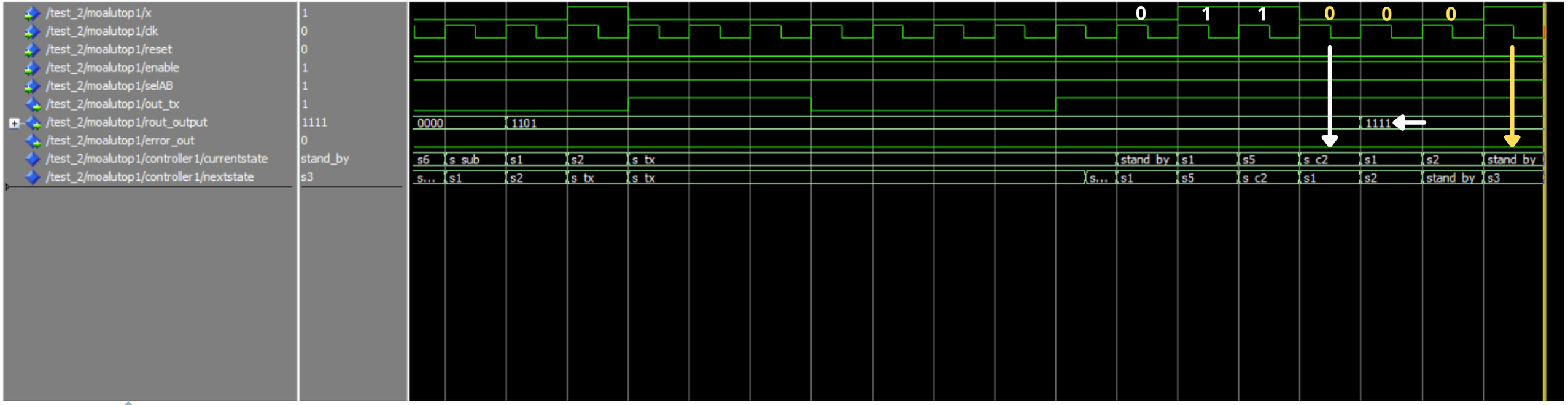
RX - SUM - TX

TEST 2 - slide 1



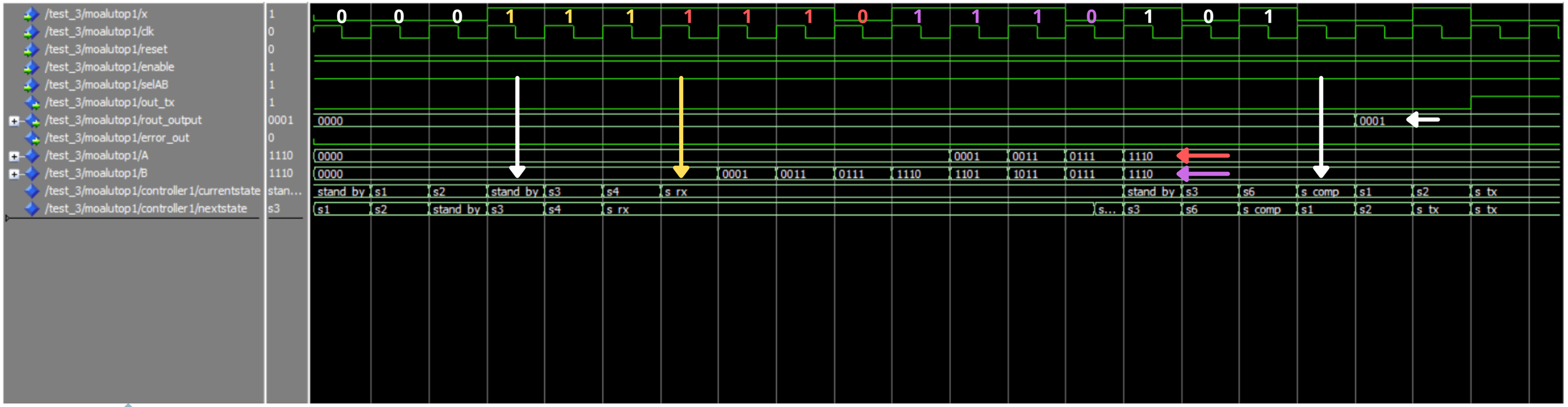
RX - SUB - TX...

TEST 2 - slide 2



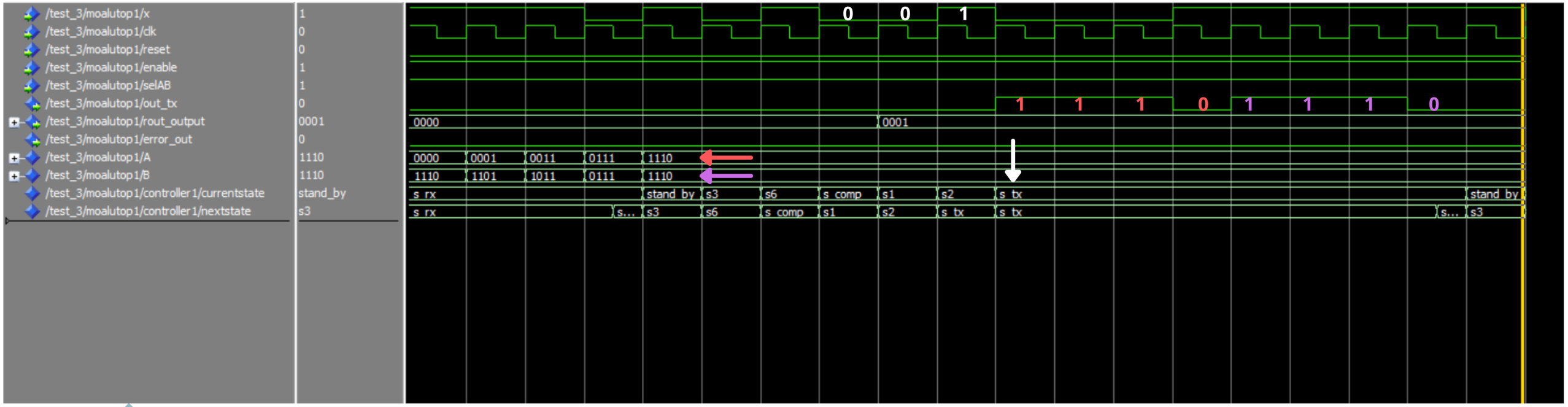
...C2 - Stand-By

TEST 3 - slide 1



Stand-By - RX - COMP...

TEST 3 - slide 2



...TX

THANKS