Byte Adder Subtractor
Quick Start

Attach the **8-bit Workbench™** *at the top edge* of the Byte Adder Subtractor to display the output of the adder. For input, use the **Binary Keyboard** or **Hex keyboard** that can attach *at the bottom edge*.

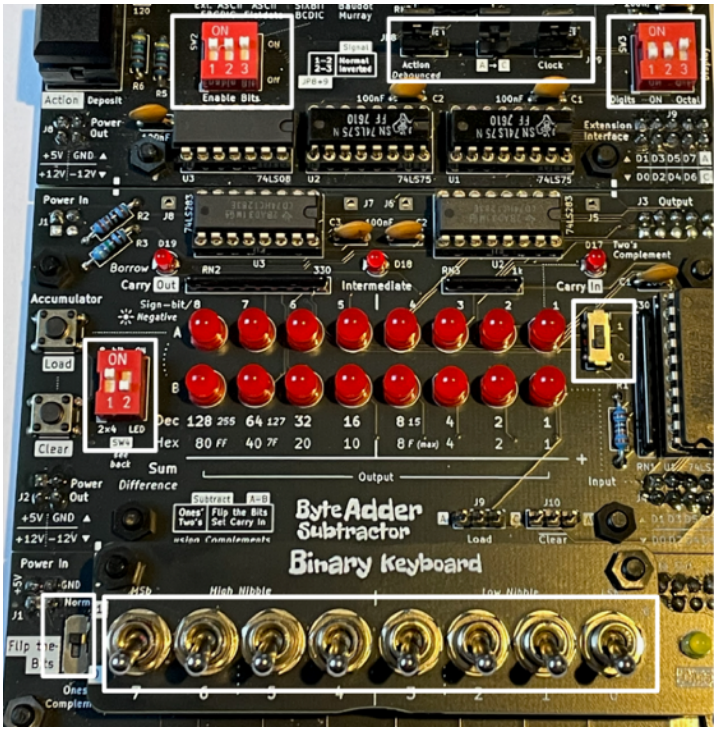
Instead, use bread boards to create your own LED output display and input mechanism, e.g. using DIP switches with pull-down resistors. *See* the schematics of our other products for inspiration how to wire this up using your own parts.

Default Setup

To follow the examples given here make sure all three boards are set with the default settings. All settings are also explained on the (back of the) board itself, so feel free to experiment.

I. For the **8-bit Workbench™**:

- Enable all most significant bits using the three DIP switches at the left hand side.
- Make sure the *Action* and *Clock* jumpers are both set in their default 'normal'-signal position: pin 1-2 is bridged (left hand side).
- Tie the *Action* to the *Clock* signal by bridging the middle jumper.
- Enable both digits and choose *Hexadecimal* (or *Octal*) display of the adder-output using the right hand side DIP switch.



II. **Byte Adder Subtractor**:

- Enable full 8-bit addition and disable the intermediate carry LED using the DIP switch.
- Slide the *Carry In* switch to zero, LED off.
- Leave out the two jumpers at the bottom side of the board.

III. **Binary Keyboard** (or **Hex Keyboard**):

- Make sure the *Flip the Bits* switch is in the normal-position, yellow LED if off.
- Set all bit toggle switches to zero, lower position (or set the hex value to 00 using the rotary switches).

Addition

First, follow the default setup outlined above. The table shows an example. To take the **sum** of two operands (**A** + **B**):

	Decimal	Binary	Hex	Octal
A	37	00100101	25	045
+ B	14	00001110	0E	016
=	51	00110011	33	063

1) After power on, press *Clear* followed by *Action* to zero all.
2) Input **A** by toggling in the binary value (or rotate for the hexadecimal value). Use the annotations on the board for easy conversions. Notice the value being updated at the lower row of LEDs, next to B.
3) Press *Load* to copy the B-input to the *Accumulator* (row A), as shown by the LEDs. Notice that now changing the input (row B) has no effect on the value that was previously stored in A.
4) Then toggle in operand **B**, again it is shown in row B.
5) Press *Action* to display the **sum** on the Workbench. In fact, the calculation is done continuously by the adder, whenever the A and B operands change. This can be visualized by removing the middle 'A → C' jumper.
6) To make another calculation: zero the input, press *Clear* followed by *Action*.

Subtraction

First, follow the default setup outlined above. The table shows an example. To take the **difference** of two operands (**A** - **B**):

	Decimal	Binary	Hex	Octal
A	103	01100111	67	147
- B	18	00010010	12	022
Ones'	*237*	*11101101*	*ED*	*355*
Two's	*238*	*11101110*	*EE*	*356*
=	85	01010101	55	125

1) After power on, press *Clear* followed by *Action* to zero all.
2) Like for addition, input **A** by toggling in the binary value (or rotate for the hexadecimal value). The value is being updated at the lower row of LEDs, next to B.
3) Press *Load* to copy the B-input to the *Accumulator* (row A).
4) Then toggle in operand **B**, again it is shown in row B. So far the procedure is the same as addition.
5) Subtraction works using complements:
   i. First take the *ones' complement* of operand B by 'flipping the bits' using the slide switch on the Binary or Hex Keyboard. This changes the B-input to the inverse, notice all the bits being flipped. Also notice the *Carry Out* LED turns on, this means the sum of the two operands result in an overflow of the byte.
   ii. Then, take the *two's complement* of operand B by sliding the carry in switch to 1. This adds 1 to the result, creating the two's complement.
6) Press *Action* to display the **difference** on the Workbench. The *Carry Out* bit can be ignored in the result.
7) To make another calculation: zero the *Carry In*, slide the *Flip the Bits* switch to normal, zero the input, press *Clear* followed by *Action*.

Notice that the result stays the same when the *ones' complement* of operand B is loaded into the Accumulator (row A) and operand A is then used as input for row B.