SYNC Signals Used to Synchronize Actions

In a (hardware) **handshaking protocol**,
- each side sends a **1-bit SYNC signal** to the other (constantly—there's no shared clock)
- and data producer sends **N bits of data** to consumer.

To send data, producer
1. **sets** the values of the **data bits**, then
2. **flips** its **SYNC bit** to indicate new data, then
3. **waits for consumer** to finish reading (at which point the consumer flips its SYNC).

SYNC Signals Combine to Obtain Status Signal

With processors and I/O devices,
- processor much faster than I/O devices
- (as students are to professors).

In LC-3 model, the **two SYNC signals are combined into a single status signal**
- with XOR in datapath (assume both bits start at 0)
- before presenting to the processor, so…
- **1 means 'ready'**
- **0 means 'not ready'**

LC-3 Changes Status Signal Implicitly with Keyboard

Consider the keyboard and the LC-3.

After the human presses a key:
1. Keyboard makes key value (in ASCII) available as data.
2. Keyboard changes status bit to 1 (ready) by flipping keyboard's SYNC bit.
3. LC-3 processor observes status bit.
4. Processor reads key value, **implicitly changing status bit back to 0** (not ready) by flipping processor's SYNC bit.

Status and Data Bits Stored in Registers

So where are the 'status' and 'data' bits?

In registers, of course!

LC-3 has 16-bit word size, so uses **16-bit registers** for convenience.

For the keyboard…
- **KBDR: Keyboard Data Register** (the key)
- **KBSR: Keyboard Status Register** (the status signal: ready or not ready?)

