



# CS4101 嵌入式系統概論

## Serial Communication

Prof. Chung-Ta King

Department of Computer Science

National Tsing Hua University, Taiwan

Materials from *MSP430 Microcontroller Basics*, John H. Davies,  
Newnes, 2008



國立清華大學

National Tsing Hua University

# Recall the Container Thermometer

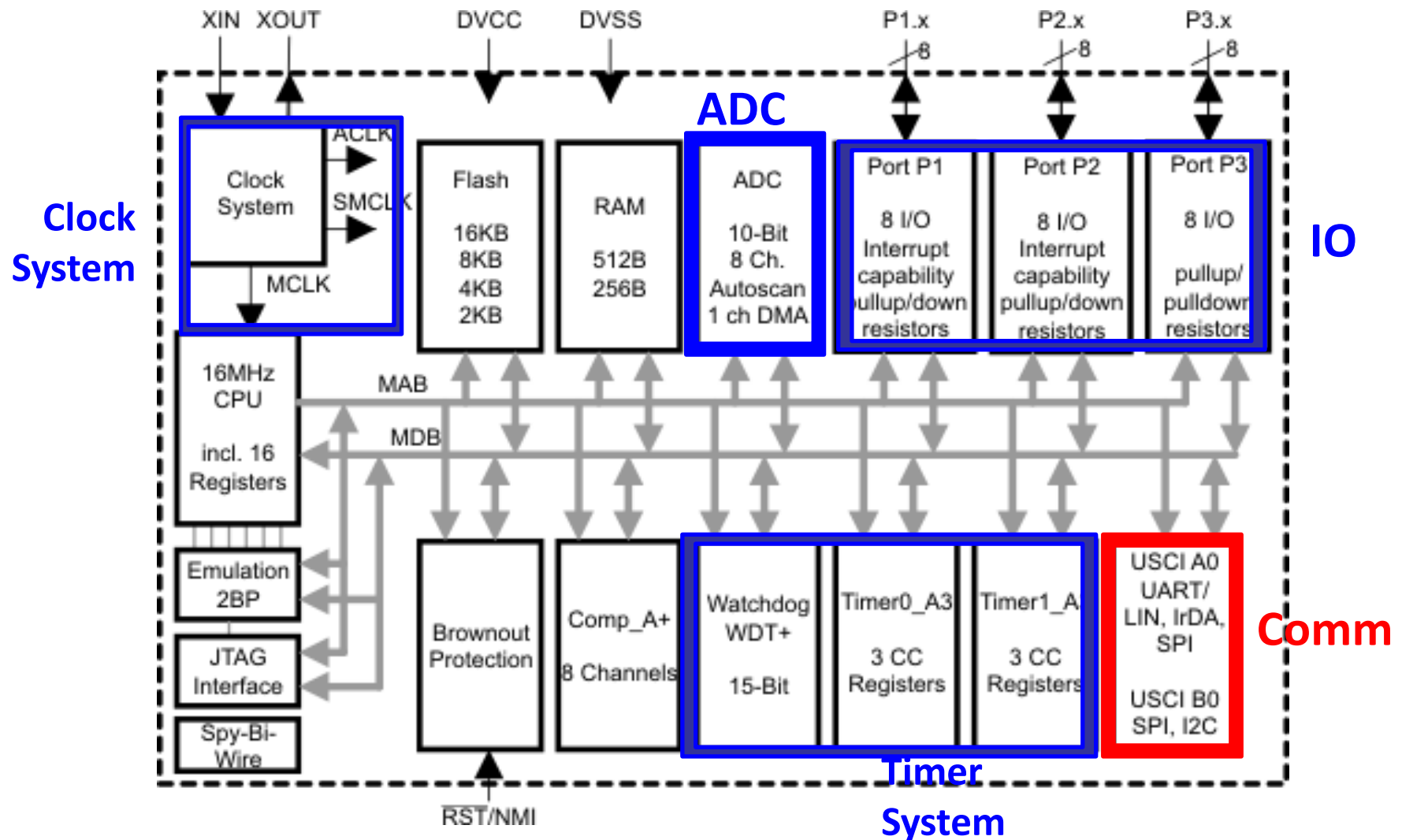
- Container thermometer: monitor the temperature of the interior of a container
  - Monitor the temperature every 5 minutes
  - Flash LED alarm at 1 Hz
  - If the temperature rises above a threshold, flash the LED alarm at 3 Hz and notify backend server
  - If the temperature drops below a threshold, return the LED alarm to normal and notify the server



**Need to communicate with the server!**



# We Have Learned ...





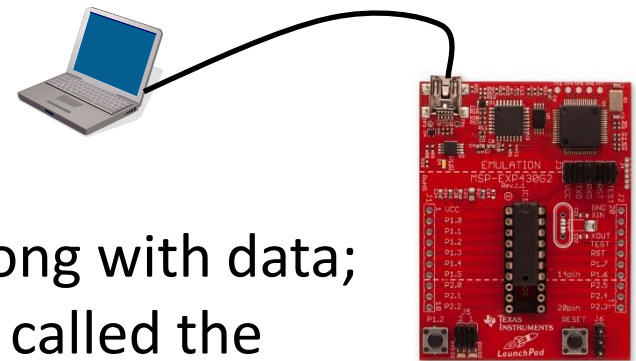
# Outline

- Asynchronous serial communication
- Asynchronous serial communication in MSP430 using software and Timer\_A



# Communication Interfaces

- Communication interfaces:
  - For exchanging information with external devices, e.g., USB, RS-232, Ethernet ...
- Serial communication:
  - A single bit is transferred at a time
  - **Synchronous**: a clock signal is sent along with data; the device that generates the clock is called the *master* and other devices are *slaves*
  - **Asynchronous**: no clock transmitted, fewer wires
- Parallel communication:
  - Multiple bits are transferred at a time





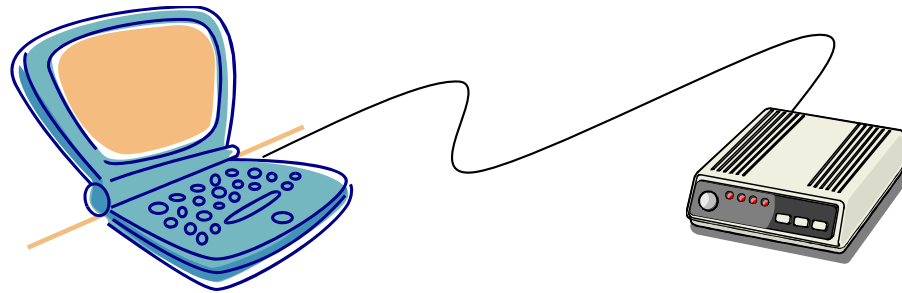
# Asynchronous Serial Communication

- Serial communication without carrying clock signal
  - Can be managed in hardware by a peripheral called a *universal asynchronous receiver/transmitter* (UART), which is built into many microcontrollers
  - Even if UART is not available, it can be emulated easily with a timer assisted by software
- Features:
  - Usually require only a single wire for each transmission direction plus a common ground wire
  - Most general-purpose connections are *full duplex*, i.e. data can be sent simultaneously in both directions
  - Often connect two end devices (not a bus)



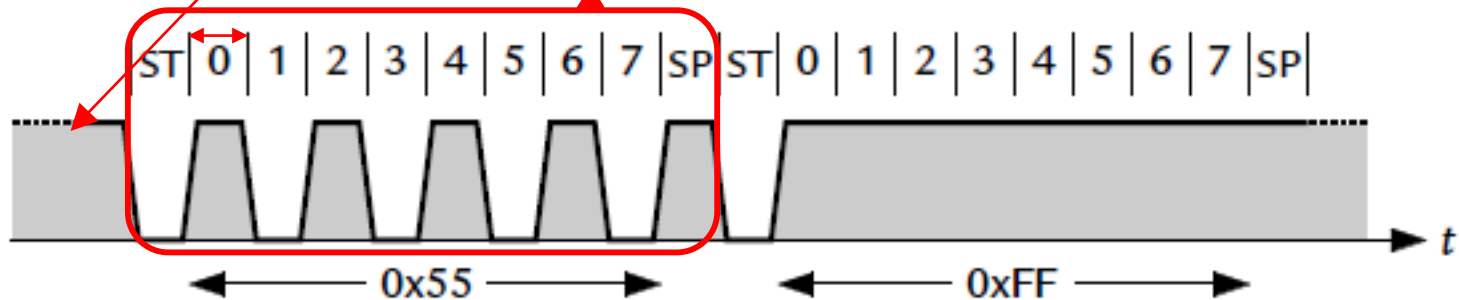
# Asynchronous Serial Communication

- How to synchronize the transmissions of the two ends which run on independent clocks?
  - Use absolute (real) time, or
  - Transmit short data (e.g. one byte) at a time, assuming the two clocks run at same rate during that period of time



# Data Format for Asyn Transmission

- Data are sent in short *frames*, each of which typically contains a single byte
- Example: the line idles high and each frame contains:
  - one low start bit (ST)
  - eight data bits, usually LSB first
  - one high stop bit (SP)




8-N-1 format: 8-bit data, no parity bit, 1 stop bit





- A standard for asynchronous serial communication
  - Originally for connecting equipment such as teletype (*data terminal equipment*, DTE) to modem (*data communication equipment*, DCE)
  - Old version, RS-232-C, published in 1969
  - Current version, ANSI/TIA/EIA-232-F, maintained by the Telecommunication Industry Association (EIA)
- Main features:
  - Connection must be less than 50 feet
  - Voltage level: 1 (-3~-15V), 0 (+3~+15V)
    - Region between  $\pm 3$  V does not correspond to valid data
    - MSP430's voltage is less than 3V  $\rightarrow$  use transceiver

Falling edge  
means 0  $\rightarrow$  1





# Transmission Speed

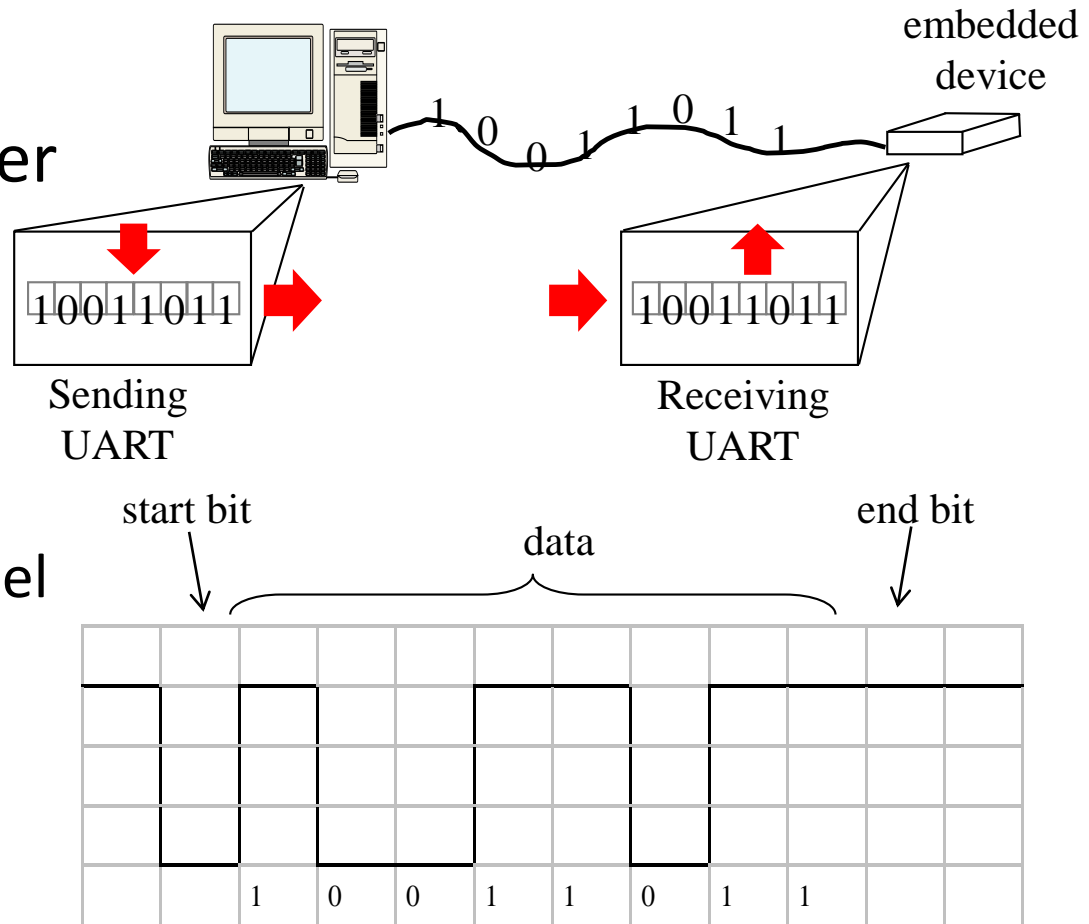
- Baud rate:
  - # of signal changes per second, e.g., 9600 baud
    - In RS-232, each signal change represents one bit, so baud rate and bits per second are equal
  - Since each 8 bits of data are accompanied by a start and a stop bit, maximum data rate is only 8/10 of baud rate
- How close must the two ends run their clocks?
  - The final sample is taken 9.5 bit periods after the initial falling edge and must lie within the stop bit
  - The permissible error is therefore about  $\pm 0.5$  bit period in 9.5 periods or  $\pm 5\%$
  - There may be errors in both receiver and transmitter, so each should be accurate to within about  $\pm 2\%$



# Serial Transmission Using UARTs

- **UART**: Universal Asynchronous Receiver Transmitter

- Takes parallel data and transmits serially
- Receives serial data and converts to parallel





# Outline

- Asynchronous serial communication
- Asynchronous serial communication in MSP430 using software and Timer\_A





# Communication Peripherals in MSP430

- Universal Serial Interface (USI):
  - A lightweight module handles only synchronous communication: SPI and I<sup>2</sup>C
  - Included in MSP430G2331
- Universal Serial Communication Interface (USCI):
  - Handle almost all aspects of the communication
  - Asynchronous channel, **USCI\_A**: act as a universal asynchronous receiver/transmitter (UART) to support the usual RS-232 communication
  - Synchronous channel, USCI\_B: handle both SPI and I<sup>2</sup>C as either master or slave
  - Included in MSP430G2553





# Software UART Using Timer\_A

- Software UART on MSP430 using Timer\_A without relying on special UART hardware
  - e.g., MSP430g2331 does not have hardware UART
  - Assume data are received and transmitted through two GPIO pins, e.g. P1.1 and P1.2, respectively
  - Use software to control Timer\_A to handle serial IO directly and process the sampled input or set up the next bit for output in an ISR





# How to Do?

- General procedure for receive (RX)
  - Hardware detect falling edge on serial input (e.g., P1.1), which indicates a start bit; start timer for 0.5 bit period
  - On timer interrupt, sample the serial input to confirm whether a valid start bit is received; if so, set timer for 1 bit period
  - On timer interrupt, sample the input to read the first bit (LSB); set timer for another bit period
  - Repeat this until all 8 bits have been received
  - Wait a further bit period and check that the input is high for the stop bit. A *framing error* occurs if this bit is low.

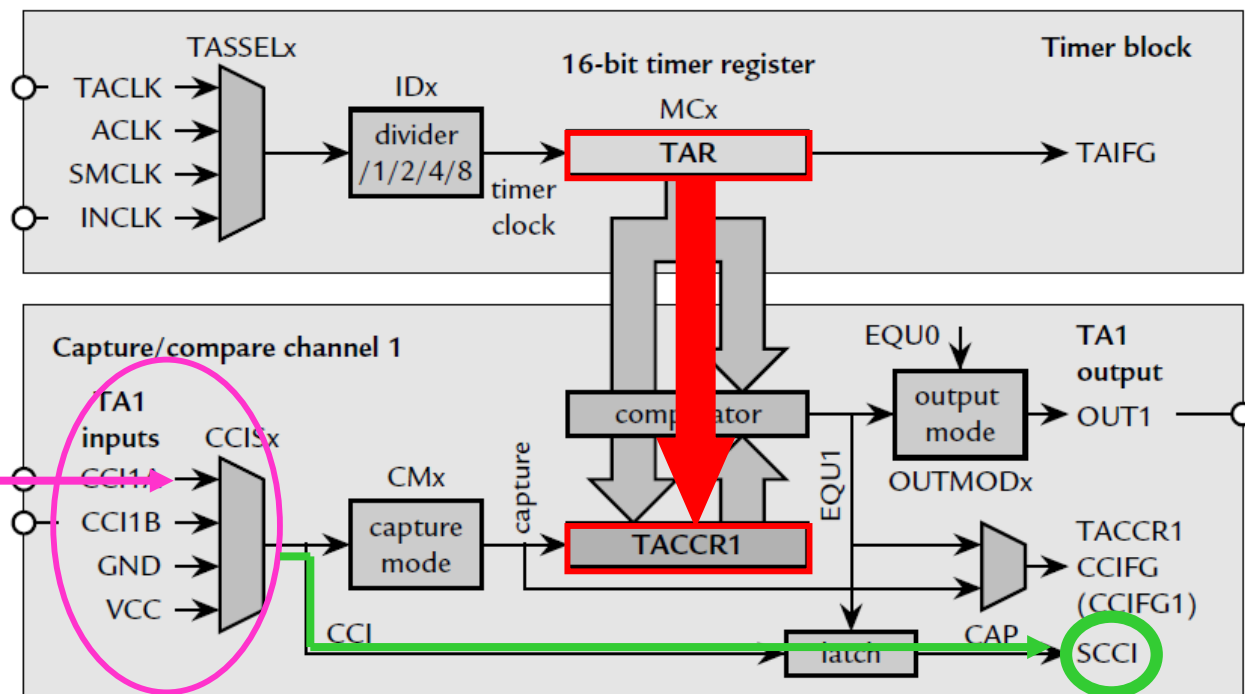
Transmission (TX) similar!



# How to Leverage Timer\_A?

- For receive:
  - Use capture mode of a capture/compare block: When an event occurs on an input to the block, the register TACCRx will store the “time”, i.e., the value in TAR, of that event
  - The input value will be latched in SCCI bit

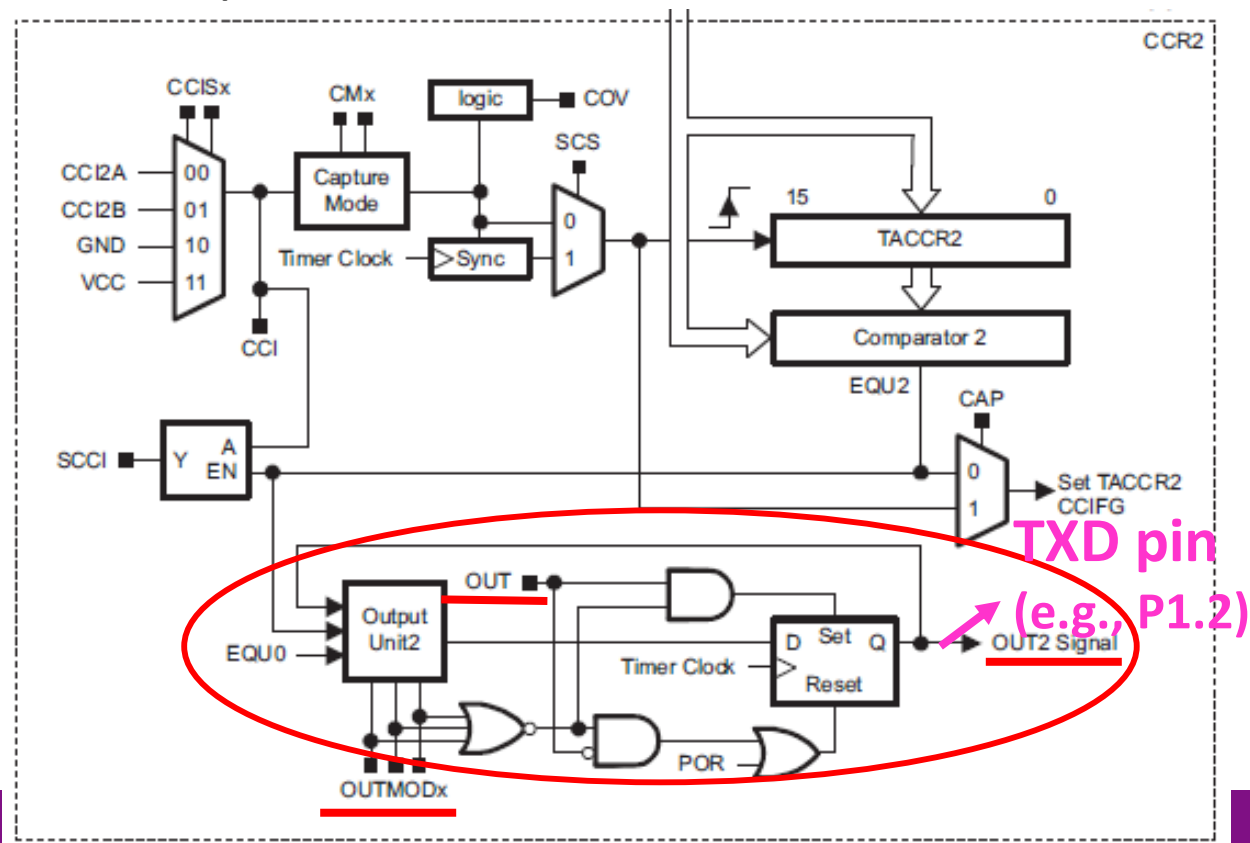
RXD pin  
(e.g., P1.1)





# How to Leverage Timer\_A?

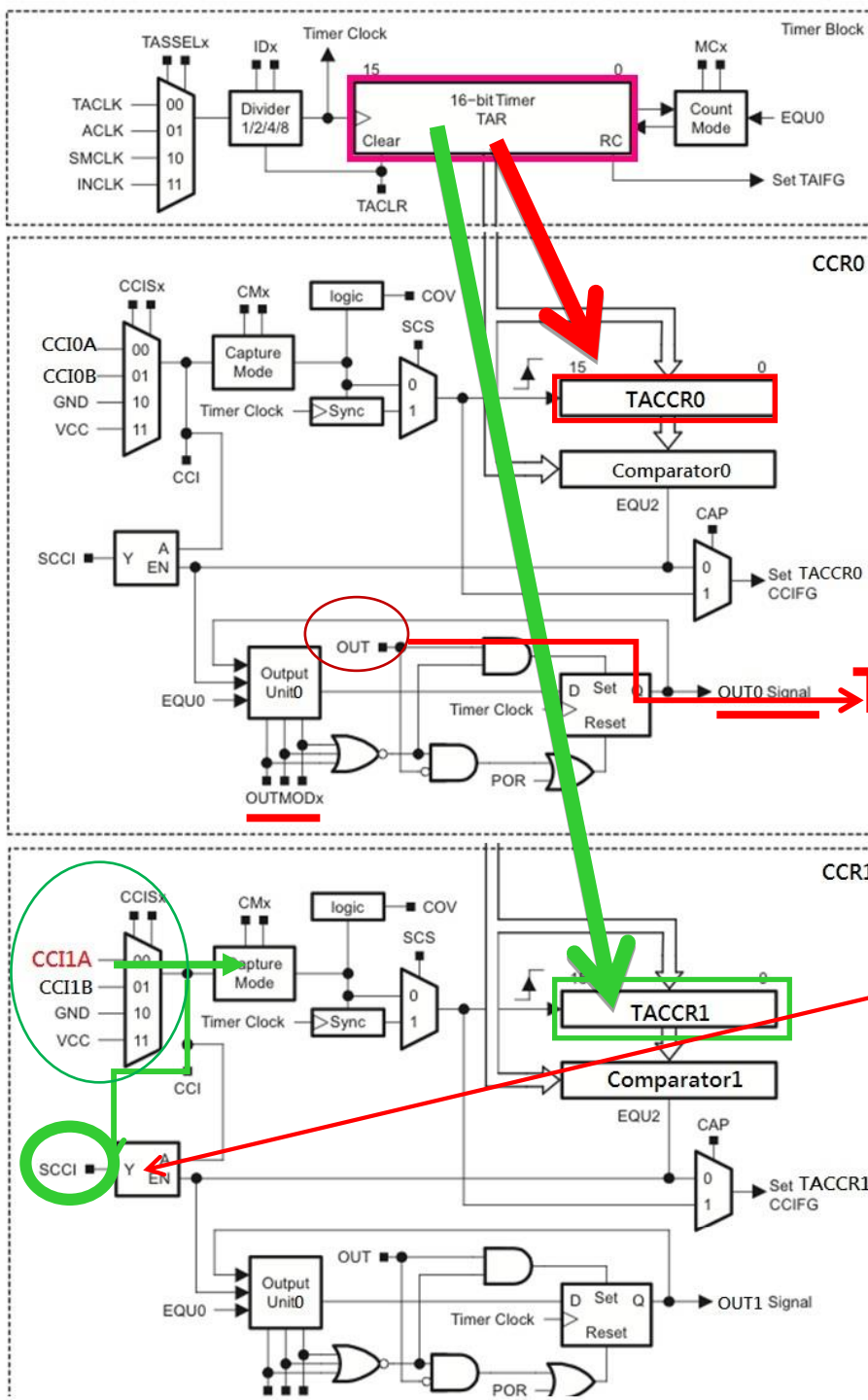
- For transmission:
  - Use output circuit of Timer\_A
  - e.g., if OUTMOD=0, output of the block is controlled directly by OUT bit in TACCTLx, as if the pin is used for normal, digital output but operated via Timer\_A



# Pin Connections

TERMINAL		I/O	DESCRIPTION
NAME	NO.		
	14 N, PW 16 RSA		
P1.0/ TA0CLK/ ACLK/ A0	2	1	I/O General-purpose digital I/O pin Timer0_A, clock signal TACLK input ACLK signal output ADC10 analog input A0 <sup>(1)</sup>
P1.1/ TA0.0/ <b>TXD</b> A1	3	2	I/O General-purpose digital I/O pin <b>Use TACCR0</b> Timer0_A, capture: CCI0A input, compare: Out0 output ADC10 analog input A1 <sup>(1)</sup>
P1.2/ TA0.1/ <b>RXD</b> A2	4	3	I/O General-purpose digital I/O pin <b>Use TACCR1</b> Timer0_A, capture: CCI1A input, <u>compare: Out1 output</u> ADC10 analog input A2 <sup>(1)</sup>

```
#define TXD 0x02 // TXD on P1.1 (Timer0_A.OUT0)
#define RXD 0x04 // RXD on P1.2 (Timer0_A.CCI1A)
P1SEL |= TXD + RXD; // Enable TXD/RXD pins
P1DIR = 0xFF & ~RXD; // Set pins to output
P1OUT = 0x00; // Initialize all GPIO
```



For  
Transmission

For  
Receive

RXD pin

TXD pin

Latch allows  
sampling  
at precise time,  
regardless of  
ISR latency

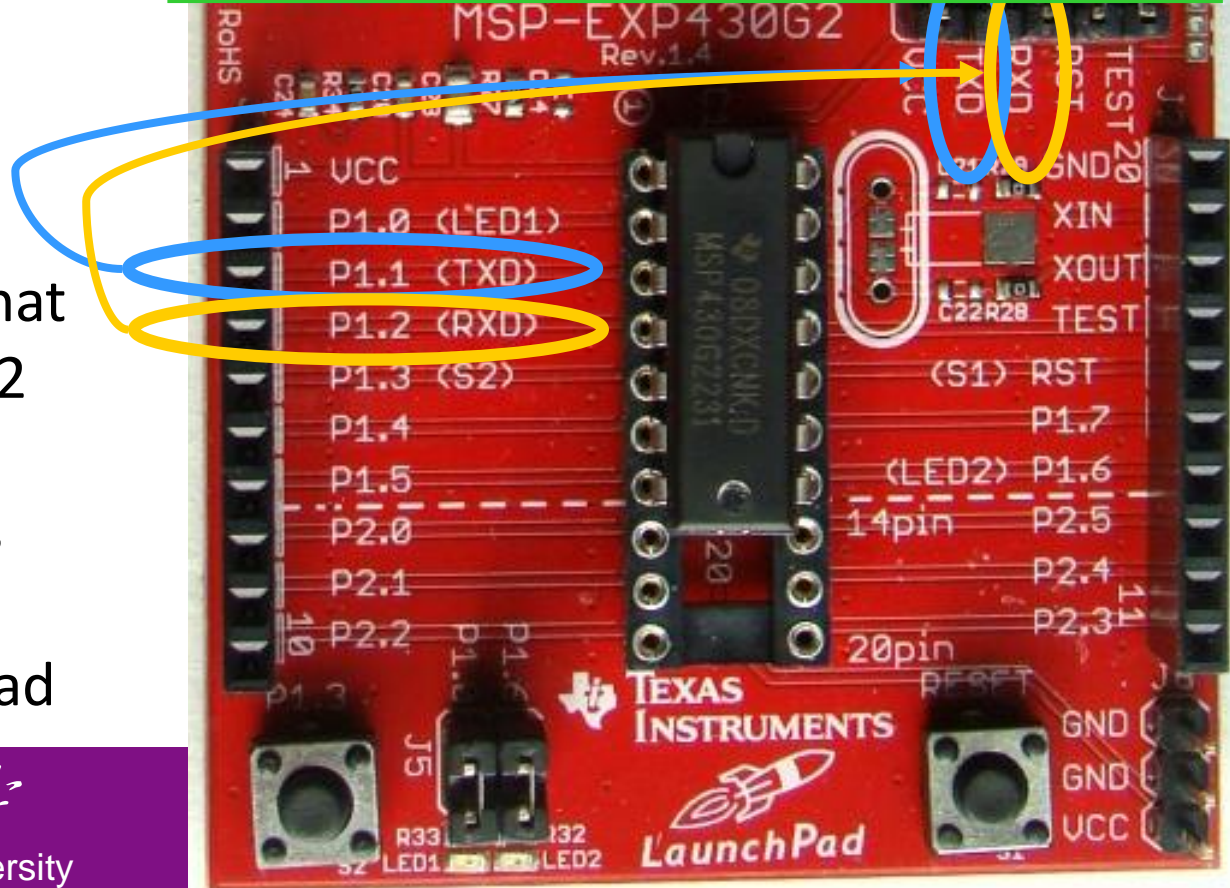


# Receive Procedure by Timer\_A

- Between transmissions, CCRx (capture/compare) waits in Capture mode for a falling edge on its input.
- When a falling edge is detected, TACCRx captures the count in TAR and an interrupt is requested. CCRx is switched to Compare mode, and TACCRx is set to fire an interrupt after 1.5 of the bit period from now.
- The next interrupt occurs and SCCI contains value of the received LSB. ISR saves it. Next compare event is set up to occur after a further bit period.
- The above procedure is repeated until all 8 bits of data have been received.



# For LaunchPad



LaunchPad thinks that it has physical RS232 links with PC, while PC also thinks it has physical COM port (RS232) to LaunchPad



國立清華大學

National Tsing Hua University