Spring 2022

ECE 568: Embedded Systems

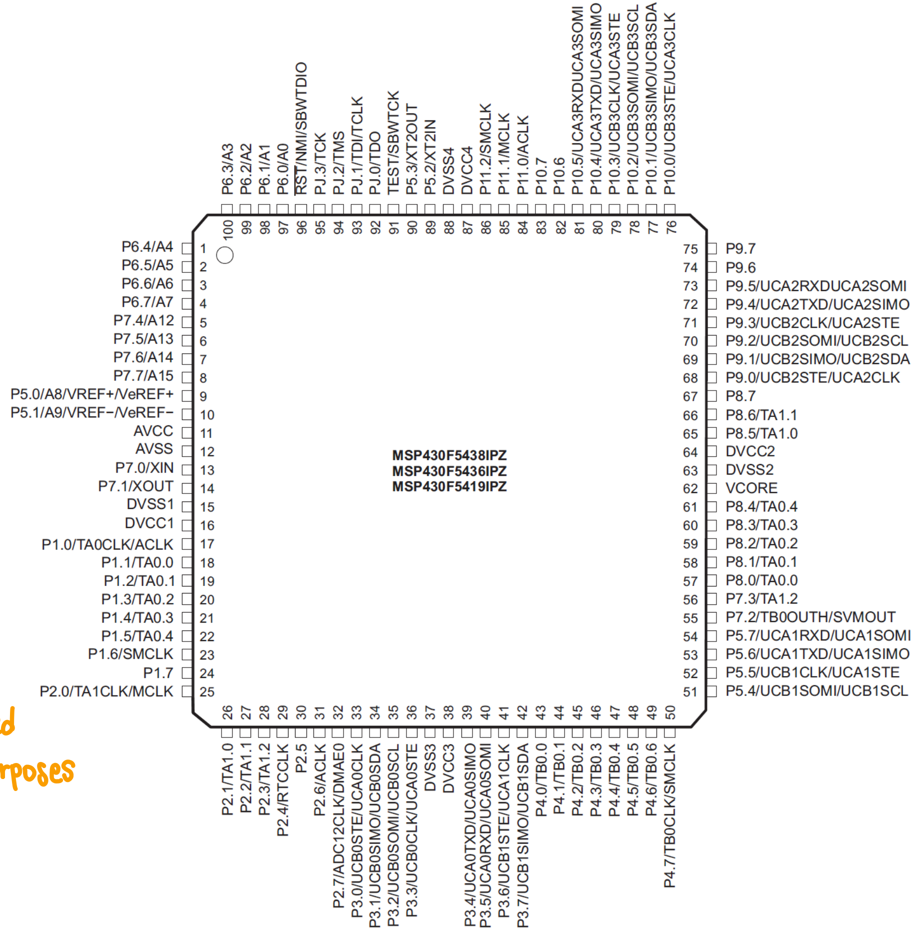
Lecture #4: Review of MCU hardware

Vijay Raghunathan

Ack.: Soumendu Ghosh

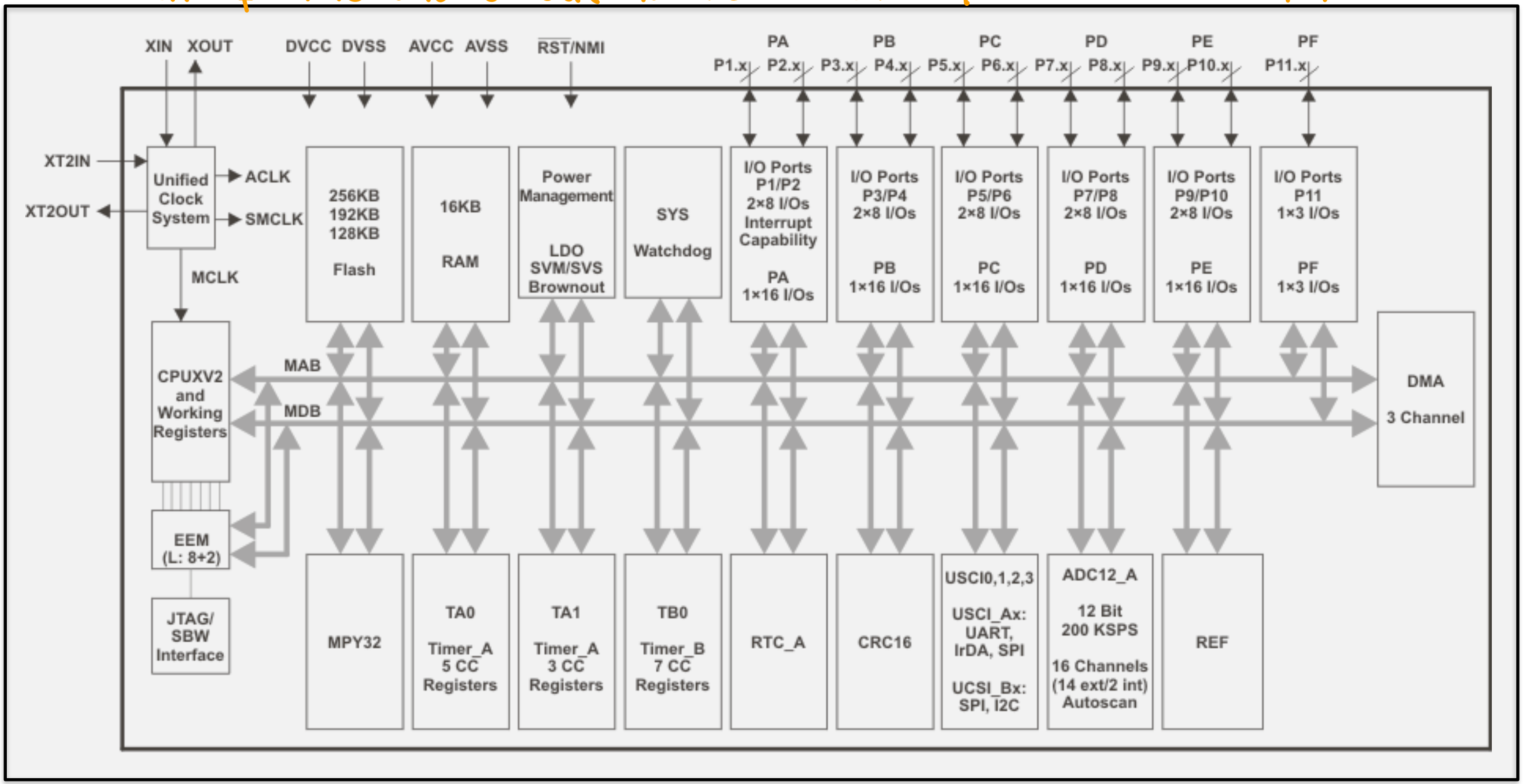
Typical MCU: Pin-Out View

- has 1000s of transistors inside
- any interaction with outside world happens using the pins
- many of the pins are very heavily multiplexed
  - pin can be used for several purposes

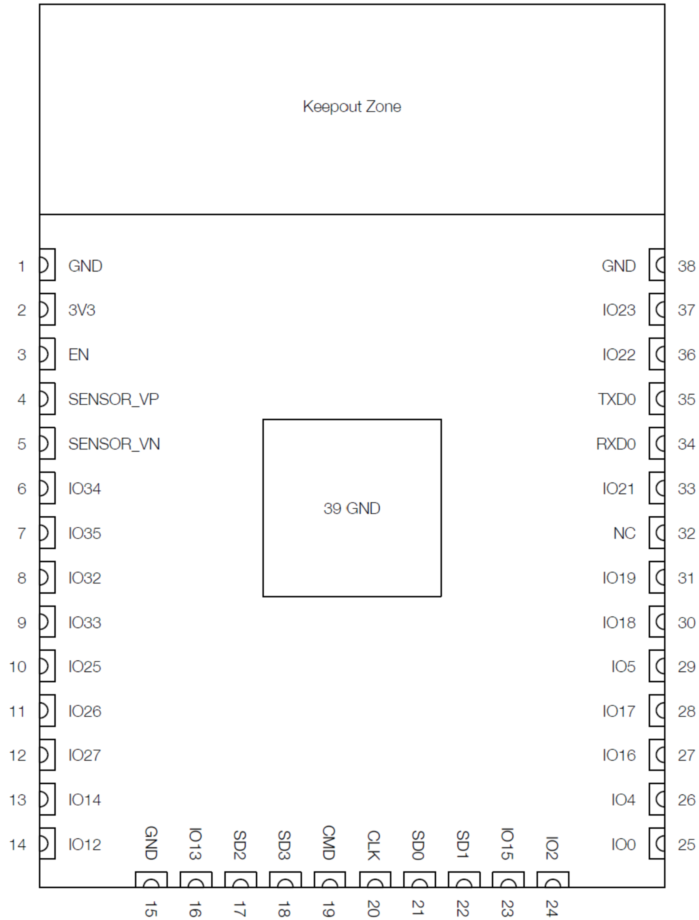


Typical MCU: Inside View

How do you build notion of time in microcontroller? How do you take the internal notion of time and correlate to the occurrence of various events in physical world?

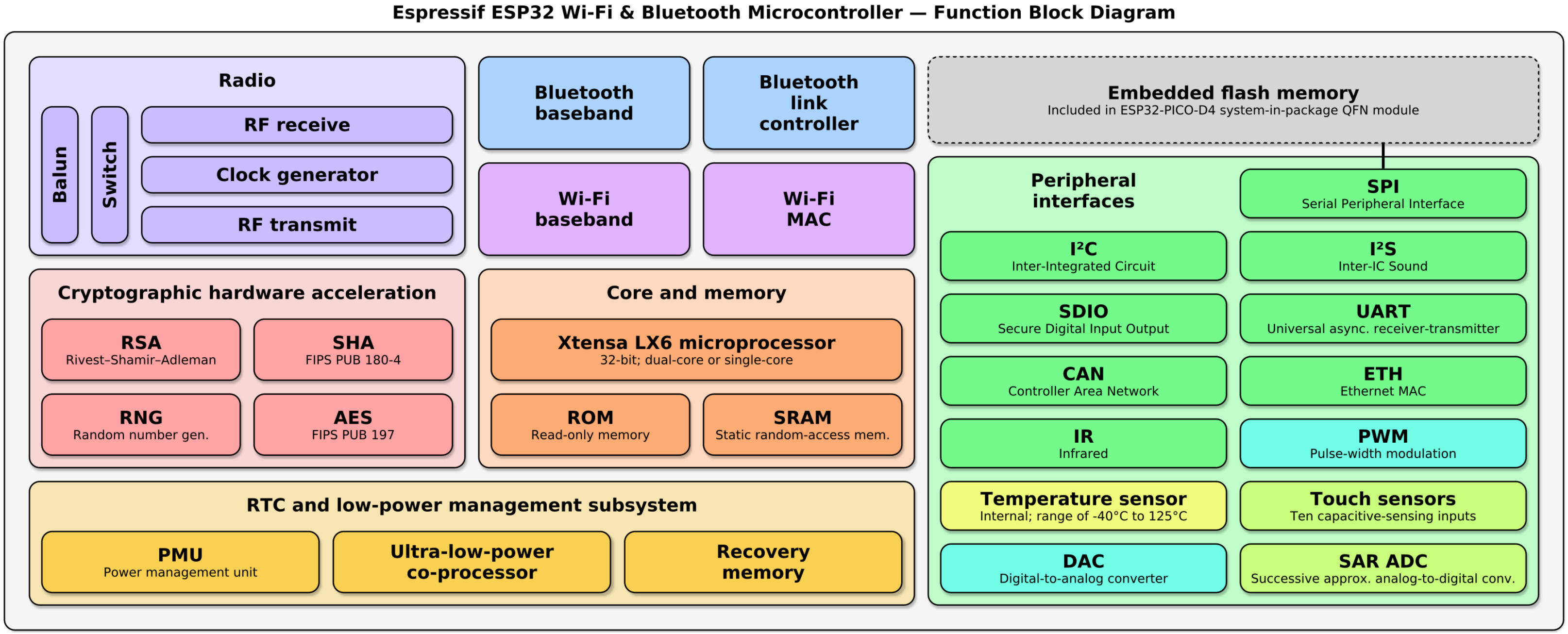


ESP32-WROOM-32D: Pin-Out View



- microcontroller has 39 pins

ESP32 Functional Blocks



[https://en.wikipedia.org/wiki/ESP32#/media/File:Espressif_ESP32_Chip_Function_Block_Diagram.svg]

Key Components of MCU

- Core, registers, MULT and CRC modules, memory map
- GPIO
- Clock sub-system
- Timers, Watchdog, RTC
- Serial Communications (UART, SPI)
- Analog to Digital Converter (ADC)
- Direct Memory Access (DMA)
- Power Management Module
- Debug and Emulation Support

ESP32 Components

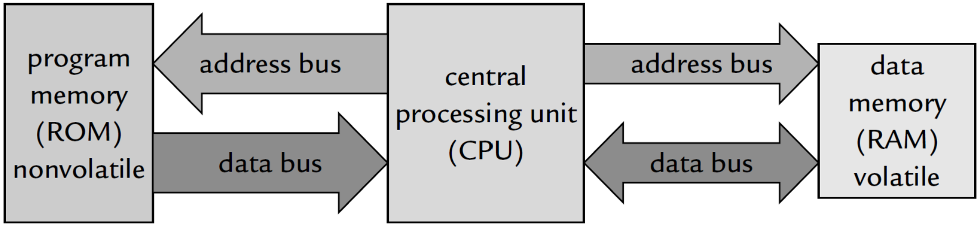
- Core and Memory
  - Single/Dual Core Microprocessor
  - Internal Memory (ROM, SRAM)
- Timer and Watchdog
  - 4 General Purpose timers
  - 3 watchdog timers (to recover from faults)
- RTC (Real-Time Clock)
- Wireless connectivity, 2.4 GHz receiver and transmitter radio
  - Wi-Fi: 802.11 b/g/n
  - Bluetooth: classic and BLE
- RTC (co-processor) and Low-Power management with multiple power modes
- Multiple GPIO pins
- Security
  - IEEE 802.11 standard security features all supported, including WFA, WPA/WPA2 and WAPI
  - Secure boot
  - Flash encryption
  - 1024-bit OTP, up to 768-bit for customers
  - Cryptographic hardware acceleration: AES, SHA-2, RSA, elliptic curve cryptography

ESP32 Components

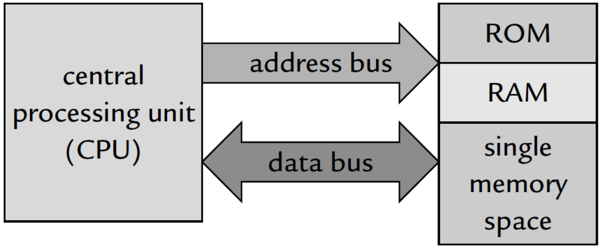
- Peripheral Interfaces
  - 12-bit SAR ADC up to 18 channels
  - 2 × 8-bit DACs
  - 10 × touch sensors (capacitive sensing GPIOs)
  - 4 × SPI
  - 2 × I^2S interfaces
  - 2 × I^2C interfaces
  - 3 × UART
  - SD/SDIO/CE-ATA/MMC/eMMC host controller
  - SDIO/SPI slave controller
  - Ethernet MAC interface with dedicated DMA and IEEE 1588 Precision Time Protocol support
  - CAN bus 2.0
  - Infrared remote controller (TX/RX, up to 8 channels)
  - Motor PWM
  - LED PWM (up to 16 channels)
  - Hall effect sensor
  - Ultra low power analog pre-amplifier
- Power management
  - Internal low-dropout regulator
  - Individual power domain for RTC
  - 5μA deep sleep current
  - Wake up from GPIO interrupt, timer, ADC measurements, capacitive touch sensor interrupt

Harvard vs. von Neumann (* very important *)

- Harvard architecture has separate instruction and data buses
  - More efficient (can read both memories in parallel)
  - Can optimize bus-width for the two memories independently
  - Many microcontrollers use this; e.g., Microchip PIC, Intel 8051, Atmel AVR, ESP32



- von Neumann architecture has a single instruction and data bus
  - Can only fetch either instruction or data on any given clock cycle
  - Microcontrollers that use this include TI MSP430, Freescale HCS



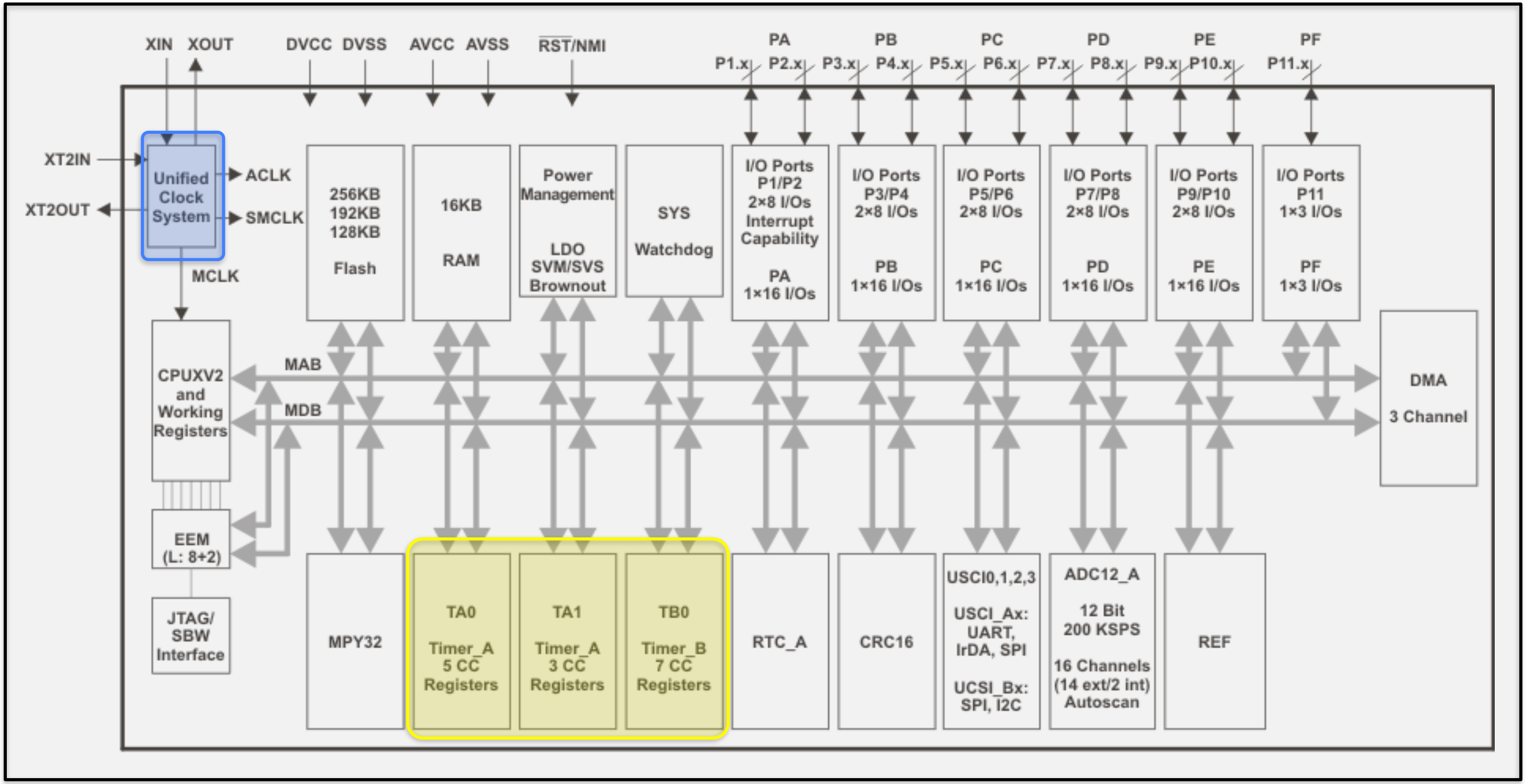
Memory Mapped I/O

- Instead of having special instructions for accessing peripheral device registers, just read/write to them as if they were memory
- Part of address space reserved for peripherals
- Device registers directly mapped to these main memory locations
- Advantage: Makes programming much simpler
  - No special instructions to access I/O devices
  - Simply used LOAD and STORE
- Disadvantage: Occupies part of memory space
  - Usually very little compared to the size of main memory
- ESP32 Address Space
  - 4 GB (32-bit) address space for both data bus and instruction bus
  - 512 KB peripheral address space

Program	45BFF 10000
Interrupt Vectors	0FFFF 0FF80
Program	0FF7F 05C00
RAM 16 KB	05BFF 01C00
Factory data (4 x 128B)	01BFF 01A00
User Info Segment A (128 B)	019FF 01980
User Info Segment B (128 B)	0197F 01900
User Info Segment C (128 B)	018FF 01880
User Info Segment D (128 B)	0187F 01800
BSL Segment 3 (512 B)	017FF 01600
BSL Segment 2 (512 B)	015FF 01400
BSL Segment 1 (512 B)	013FF 01200
BSL Segment 0 (512 B)	011FF 01000
Peripherals 4 KB	00FFF 00000

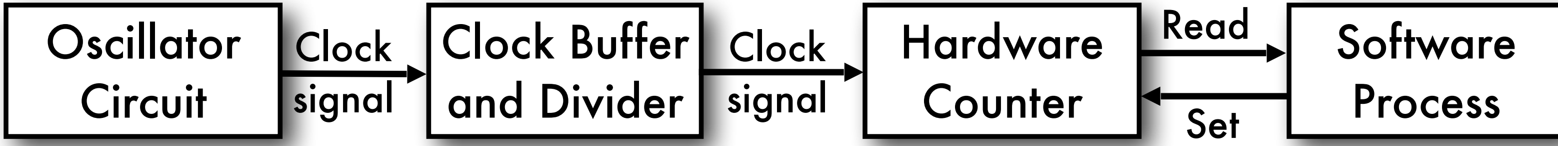
Clocks in MCU

1. How does a MCU get the notion of time? Where/How does it know?



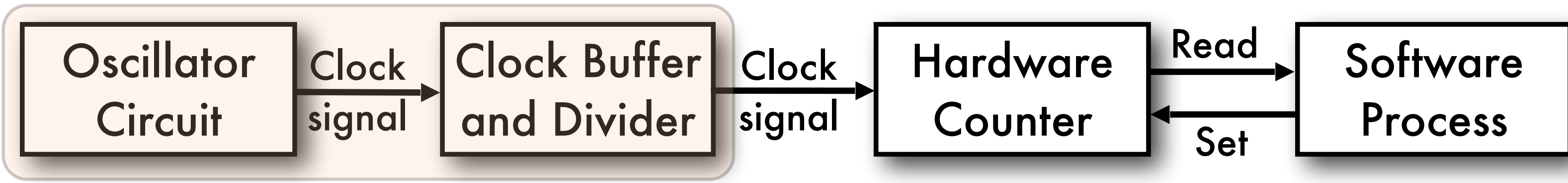
2. How do you take this MCU time and correlate with outside events?

Time in Embedded Systems



- Time is kept by a hardware counter that is fed by a clock signal
  - The hardware counter is n bits wide; counts from 0 to $(2^n - 1)$ and rolls over
  - The clock signal has a frequency f_{CLK}
- The clock signal increments the counter every 1/f seconds (resolution)
  - At time t, the counter reads $c(t) = \lfloor f_{CLK} \times t \rfloor \bmod 2^n$
- Software can read the counter or set it to a particular value
  - Smallest increment at which software can read the counter (precision)
- How close is timer to UTC? (accuracy)

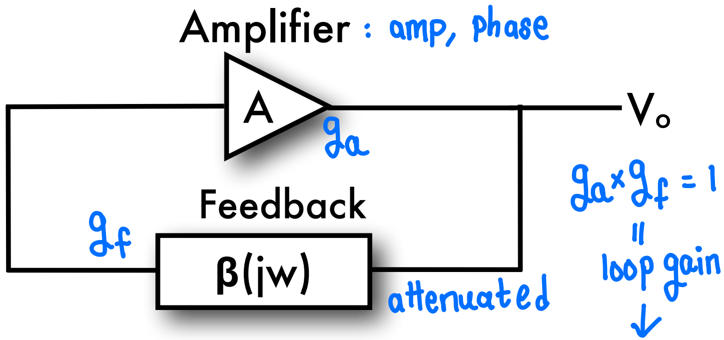
Clock Generation

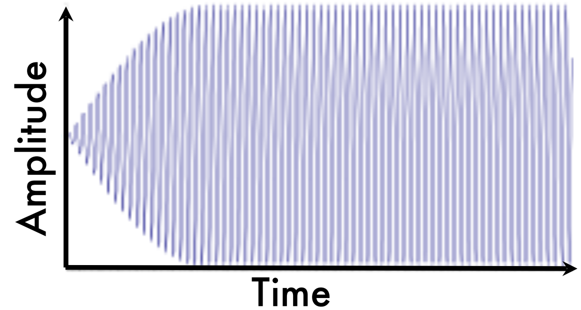


- An electronic oscillator is a circuit that produces a periodic electronic signal, often a sine wave or a square wave
- Most oscillators operate using the notion of positive feedback
- Several varieties exist. Common types in embedded systems include:
  - Ring Oscillators
  - RC Circuit Oscillators
  - Quartz Crystal Oscillators
  - Ceramic Resonator Oscillators

Should know what these are!

Oscillator Basics





- Barkhausen criterion: A linear circuit with a feedback loop will sustain steady-state oscillations only for frequencies at which the loop gain (product of forward gain and feedback gain) has a magnitude equal to one and a phase shift equal to zero or an integer multiple of 2π
- In effect, feedback network acts as a band-pass filter
  - Initially, only noise present at amplifier input; feedback network only allows the oscillation frequency through, amplitude increases till stable oscillations start