FIFO

goal: no delay, apple, SW needed to ground entire string out UART

Non-blocking

while (1)

switch (state) {
  case 1:
  ...:

application: count, ad "eat" results

RTC - update SW RTC

Timer - send time

UART

RX - r, r, v: switch state
    d/R: charge status

TX - take next byte from FIFO
    cmd: UART

Count
Multi-tasking:

- editor
- compiler
- itunes
- youtube

How is each in a process, feels like it owns the machine.

4 GB address space
2 56 MB for user memory mapped

1 MB program?
1MB data / stack heap

can have many programs in memory

Conceptually:

- every 500,000 cycles, INT
- OS stores EPC, R[0:31]
  - CPI (CP1) ret
- OS stack ~ 12KB
- loads new EPC, R[0:31]
  from OS stack
  
- ret
- 500 cycles? probably less

0.1% overhead
10ms chunks
Memory protection

User who controls MPU registers?

MMI/O?

Kernel

Kernel/User bit.

Boot, INT - Kernel mode
Ripple Carry Adder $A_{31}$

$$
\begin{align*}
\text{delay} & = 32 \times t_{	ext{ci/co}} \\
& \approx 1\text{ns} \text{ on Xilinx}\end{align*}
$$

$$
\begin{align*}
t_{	ext{ci/co}} & = t_{\text{AND}} + t_{\text{OR}} \\
& \approx 30\text{ps} \text{ on Xilinx}\end{align*}
$$

Subtraction

"Spider Kids game"
roll a 7 ⇒ go back 1
"111" = "-1"

if all you have is an adder, how do you subtract?

\[ A \rightarrow -A? \]

\[ A + \neg A = \text{all } 1s \]

So \[ A + (\neg A + 1) = \text{all } 0s = 0 \text{ (plus carry)} \]
how often do we get word one delay and if odd,? whenever the result is 0?

How to reduce?

more complex gates

16 t

2\times 2\times 2\times 2

but bigger total

make each 16 bit adder out of 2 8's

1.5 x again \Rightarrow (1.5)^2 area, power

8 t_{cilo} + 2 t_{mux}

make 8^2 out of 4's

(1.5)^3 area, power

8 t_{cilo} + 3 t_{mux}

c_{i} = a_{i} \cdot b_{i}
\rho_{i} = a_{i} \oplus b_{i}
Carry Lookahead:

\[ C_3 = G_3 + \]
\[ G_2 P_3 + G_1 P_2 P_3 + G_0 P_1 P_2 P_3 \]

Carry Bypass - middle fast/small/Merger:

- All GPs show up at same time.
- If there is a carry, needs to ripple through the 4 MSBs, then through the Mers.

Worst case: 4 carries in a row;
propagate the whole way.