Let’s summarize what we’ve learned about controlling pipelined systems.

The most straightforward approach is to use a pipeline with the system clock chosen to accommodate the worst-case processing time.

These systems are easy to design but can’t produce higher throughputs if the processing stages might run more quickly for some data values.

We saw that we could use a simple handshake protocol to move data through the system.

All communication still happens on the rising edge of the system clock, but the specific clock edge used to transfer data is determined by the stages themselves.

It’s tempting to wonder if we can might adjust the global clock period to take advantage of data-dependent processing speedups.

But the necessary timing generators can be very complicated in large systems.

It’s usually much easier to use local communication between modules to determine system timing than trying to figure out all the constraints at the system level.

So this approach isn’t usually a good one.

But what about locally-timed asynchronous systems like the example we just saw?

Each generation of engineers has heard the siren call of asynchronous logic.

Sadly, it usually proves too hard to produce a provably reliable design for a large system, say, a modern computer.

But there are special cases, such as the logic for integer division, where the data-dependent speed-ups make the extra work worthwhile.

We characterized the performance of our systems by measuring their latency and throughput.

For combinational circuits, the latency is simply the propagation delay of the circuit and its throughput is just 1/latency.

We introduced a systematic strategy for designing K-pipelines, where there’s a register on the outputs of each stage, and there are exactly K registers on every path from input to output.

The period of the system clock t_CLK is determined by the propagation delay of the slowest pipeline stage.

The throughput of a pipelined system is 1/t_CLK and its latency is K times t_CLK.

Pipelining is the key to increasing the throughput of most high-performance digital systems.