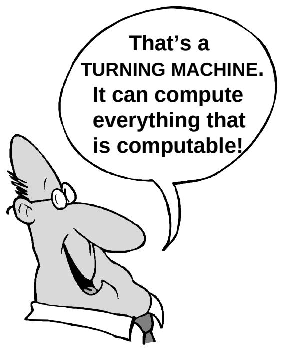
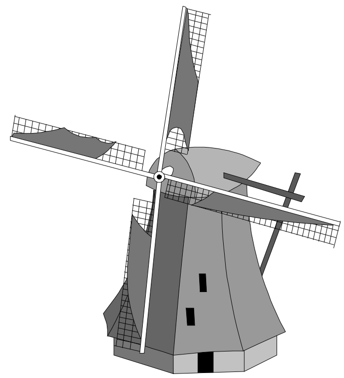
6004 Spring 1998: L26

Turing Machines and Sequential Interpretation





1

Recap: Control and Datapaths

- **Can use different control FSMs with the same datapath to compute different functions**
- **The control structure we consider is <u>synchronous globally timed</u> where a centralized clocked FSM generates all signals**
- **Rather than writing state tables to describe each control FSM, we can write RTL or machine language**

- **Is there a universal datapath or set of datapaths that can compute all computable functions?**

2

Unbounded Space Computations

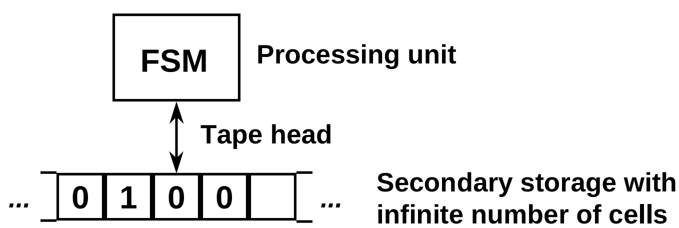
- **Cannot construct an FSM that checks if an input string is of the form $a^n b^n$, n > 0 (e.g., ab, aabb, …)**

After processing the a symbols in the sequence, we need to remember the number of a symbols to check that there are an equal number of subsequent b symbols

A <u>finite</u> state machine cannot remember an unbounded quantity!

3

Turing Machine



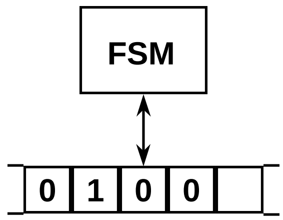
- **2-way infinite tape, discrete symbol positions**
- **Finite alphabet e.g., {0, 1, a, b, x, y, □}**
- **FSM reads and writes symbols**
  - **Input: current symbol**
  - **Output: write symbol, move left or right**

Blank symbol

4

6004 Spring 1998: L26

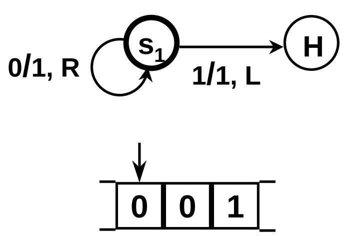
Turing Machine Operation



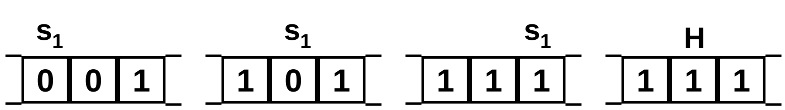
- Machine starts in given initial FSM state with initial tape configuration
- Sequence of steps is controlled by FSM
  - Contents of any cell can be examined or changed many times
- Eventually the whole process may or may not terminate. If it does machine is said to <u>halt.</u>

5

Turing Machine Operation Example - I



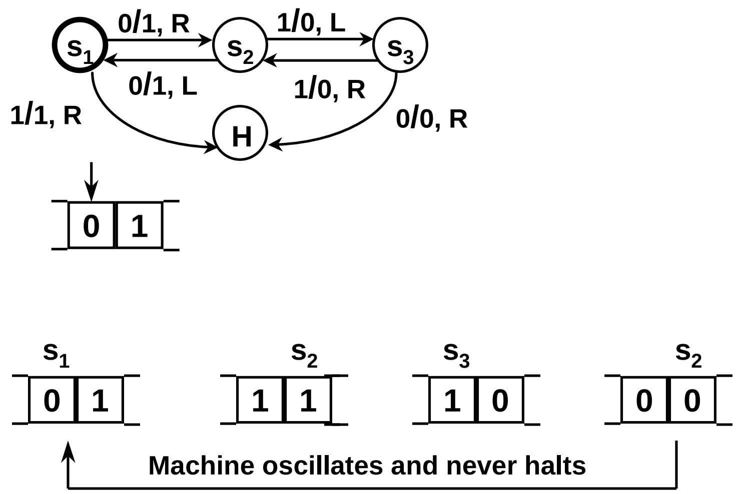
Sequence of FSM states and tape configurations



Machine halts

6

Turing Machine Operation Example - II



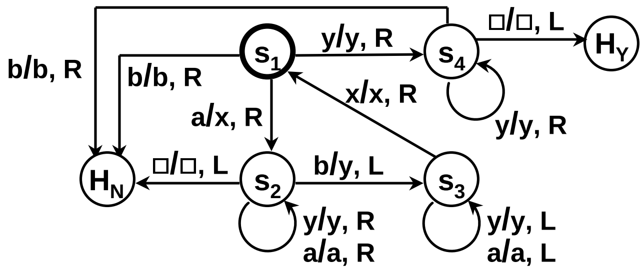
7

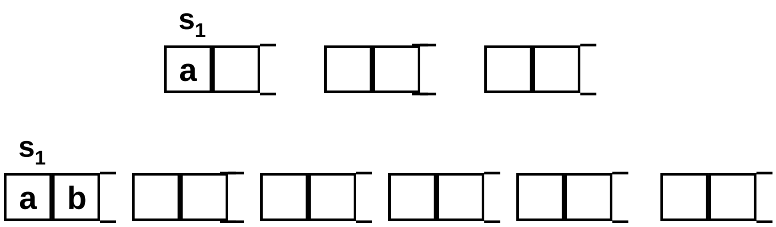
Algorithm for Checking $a^n b^n$ Form

- **Start at leftmost a**
  Replace with symbol x
- **Travel right to find leftmost b**
  Replace with symbol y
- **Travel left to leftmost a**
  Replace with symbol x
- **Terminate if you encounter a ☐. Upon termination, direction of movement and whether you are searching for an a or a b determines if string is of the form $a^n b^n$.**

8

Turing Machine for $a^n b^n$ Checking ($n > 0$)





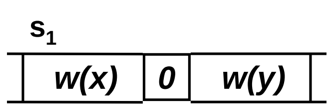
9

Computing $x + y$

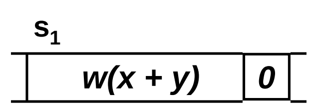
Represent positive integers, *x* and *y* in unary notation

x	*w(x)*
1	*1*
2	*11*
3	*111*

Initial tape configuration



Final tape configuration



All we need to do is to move the separating *0* to the right of *w(y)*

10

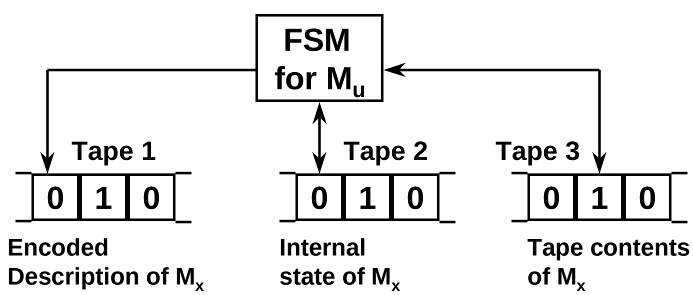
Church-Turing Thesis

Hypothesis: Any computation that can be carried out by mechanical means can be performed by some Turing machine.

We could conceivably implement the β instruction set on a Turing machine !

11

Universal Turing Machine (UTM)



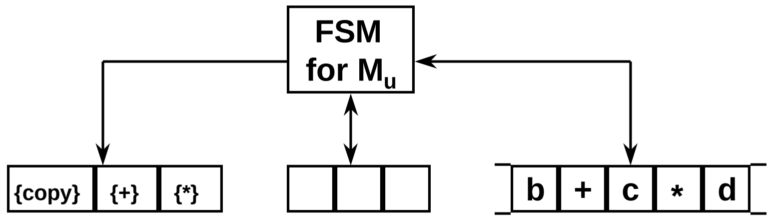
First looks at contents of Tapes 2 and 3 to determine configuration of M_x.

Then consults Tape 1 to see what M_x would do in this configuration.

Finally, Tapes 2 and 3 will be modified to reflect the result of the move.

12

UTM Viewed as an Interpreter



Description of Turing machine FSMs for copy, +, *

States of FSMs for copy, +, *

Input data y, Instruction sequence P
Output results

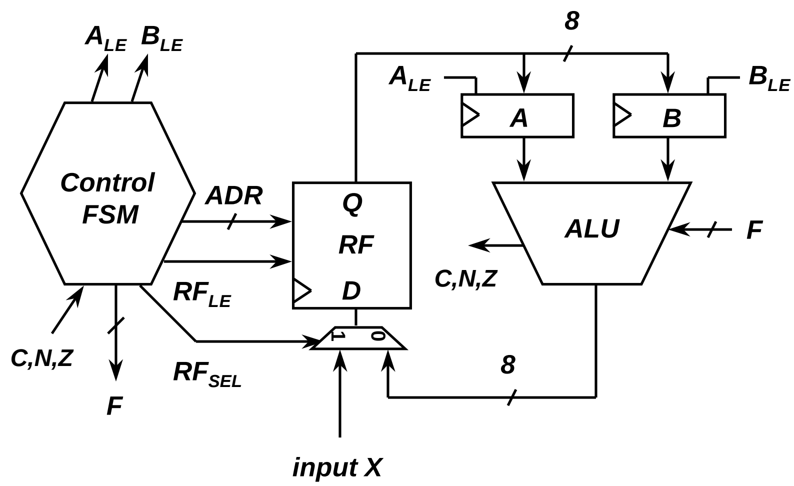
Interpreter → M_u[{x1}, y] = M_{x1}[y] ← data

.
.

Microcode written ONCE → M_u[{xn}, y] = M_{xn}[y] ← Specific tasks or instructions

13

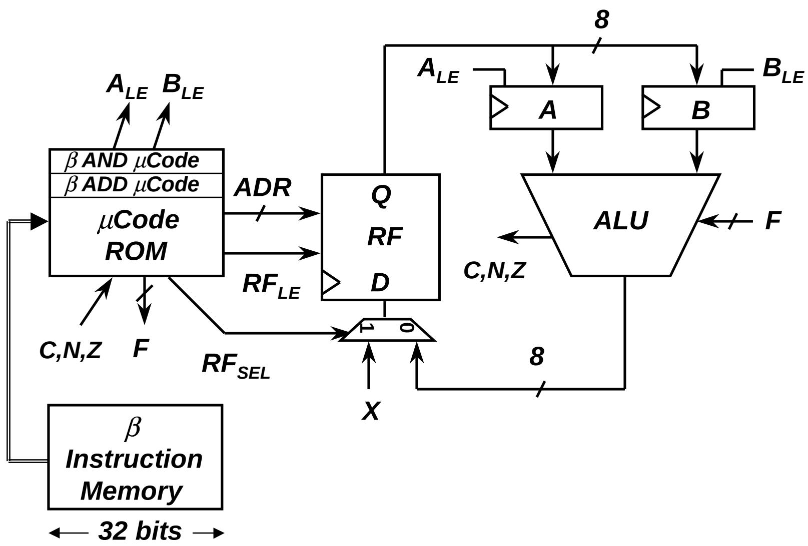
8-Bit "Toy" Datapath



14

6004 Spring 1998: L26

β on Toy



15

μCode for β AND

AND(r0, r1, r2)

RF 0-3 RF 4-7 RF 8-11

A ← <RF0>
B ← <RF4>
<RF8> ← <A> & <B>
A ← <RF1>
B ← <RF5>
<RF9> ← <A> & <B>
A ← <RF2>
B ← <RF6>
<RF10> ← <A> & <B>
A ← <RF3>
B ← <RF7>
<RF11> ← <A> & <B>

16

μCode for β ADD

ADD(r0, r1, r2)

RF 0-3 RF 4-7 RF 8-11

```
A ← <RF0>
B ← <RF4>
<RF8> ← <A> + <B>
if (C == 1) {
   A ← <RF1>
   B ← <RF5>
   <RF9> ← <A> + <B> + 1
} else {
   A ← <RF1>
   B ← <RF5>
   <RF9> ← <A> + <B>
}
if (C == 1) {
   A ← <RF2>
   B ← <RF6>
   <RF10> ← <A> + <B> + 1
} else {
   A ← <RF2>
   B ← <RF6>
   <RF10> ← <A> + <B>
}
if (C == 1) {
      ....  RF11 ....
}
```

N.B. C is combinational, only valid during one clock cycle!

17

β Instruction Dispatch

- **Do we write different μCode for each possible *ADD(r[0-31], r[0-31], r[0-31])* ?**
  ra rb rc

- **Alternate strategy:**
  - **Based on ra field, copy appropriate register contents to designated temporary registers T0-3**
    - » **Do this using 32-way switch statement**
  - **Based on rb field, copy appropriate register contents to designated temporary registers T4-7**
  - **Compute T0-3 + T4-7 in T8-11**
  - **Based on rc field, copy T8-11 to appropriate β register**

18

Interpretation: Practical Benefits

- **Bootstrap high-level functionality on very simple hardware**
- **Deal with representations of machines rather than machines themselves**
  - **Representation is now a mathematical object**
  - **Can write compilers to automatically translate between different representations**

Many real machines are built using several interpretive layers

19