

A RAILROAD SWITCHING YARD CONTROLLER: DIGITAL ELECTRONICS AS AN EDUCATIONAL TOOL

I. Program Description

A 6-week Inner-city Commuter Intervention Program for
at-risk Latino 7th - 9th grade students

- A. Meets from 8:00 to 12:30, 5 days a week, 13 weeks in fall
(170 contact hrs.)
- B. Content Focus in CS, DL, Engineering
- C. Program is *bilingual*
- D. Students *assemble the computers*
- E. Integrated into *pipeline of 16 area programs* (Access 2000)
- F. Statistics
 1. **194** Latino students over the last 4 years.
 2. **2:1** applicants to available spots
 3. **Gender balanced**: over 50% have been women.
 4. **50%** have completed the 13 week academic year followup
 5. **20%** received high school course credit.
 6. **100%** indicate increased interest in SMET
- G. Funding: NSF, DOE, CPS (~\$45K / yr.)

A Sample Lesson: Train Switching Yard Design

BUILDING A RAILROAD SWITCHING YARD CONTROLLER: DIGITAL ELECTRONICS AS AN EDUCATIONAL TOOL

Dale Reed[†]

Digital electronics can be used as an educational tool to show students how technology can be applied in every-day situations. Implemented in a bilingual fashion for inner-city *middle school students*, we explore digital electronics as an entry point into understanding how computers work, covering material normally not encountered until the university level. The presentation of this material assumes *no previous background* in engineering or computer science. Students design and build circuits that could be applied in real-life around them. Taking the projects from design to actual implementation using locally available inexpensive materials, digital electronics functions as a technology that motivates the students to understand science, math, and engineering concepts. The six-month program is funded by the National Science Foundation (NSF).

Following is a brief outline of the process we go through in teaching students how to create digital electronic designs. Students begin with an introduction to boolean logic as well as base two numbers. We then “re-write” the boolean expressions using symbols, and then again using digital gate diagrams. Circuit designs are represented in truth-tables, which are simplified using a technique called K-maps. This design process is then applied to building a railroad switching yard controller.

STEP 1: Base 10 numbers, Base 2, TRUE or FALSE

The meaning of “base 10” is explored, after which we extrapolate to base 2 (binary) numbers. Boolean logic (Fig. 1) is introduced, where we interchange TRUE/FALSE with 1/0. All the possibilities of the boolean operators are then shown using truth tables (not shown here).

- | | |
|---|-------|
| 1. We are Alive | |
| 2. We are Dead | |
| 3. (We are Alive) AND (We are Dead) | [AND] |
| 4. (We are Alive) OR (We are Dead) | [XOR] |
| 5. We are NOT dead | [NOT] |
| 6. (1 min. = 60 sec.) OR (a day = 24 hrs) | [OR] |

Fig. 1: Boolean Logic Operator Examples

STEP 2: Using gates, adding in Binary, Designing a half-adder.

Next we practice evaluating statements containing symbols for OR (+), XOR (\oplus), AND (\bullet), and NOT (\neg), as shown on the left of Fig. 2. We transcribe these into digital gate diagrams (on right of Fig. 2). Next we review

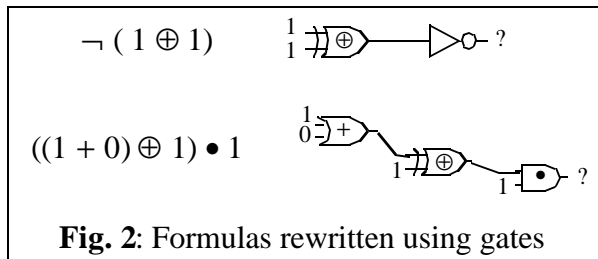


Fig. 2: Formulas rewritten using gates

[†]University of Illinois at Chicago, EECS Department, 851 South Morgan Street (MC 154) Chicago, Illinois 60607. email: reed@eeecs.uic.edu, Tel: (312) 996-3423

addition in base ten, where the teacher pretends to not understand how to add in base ten, getting the students to explain it (e.g. use a “carry” when we go over 10 in one column). We then use the same principles to do it in base *two*! (Fig. 3) Then we show how we can represent the relationship between inputs and outputs of a circuit by using a truth table. In Fig. 4 we develop the truth table for all the possibilities of adding 2 single *binary digits (bits)*. Formulas for the outputs (*x* and *c*) are derived from the truth table, and the gate diagram is derived from the formulas. At this point, amazingly enough, we have designed the circuit to add two single-digit binary numbers, *using only gates*! Students are surprised to see that they can “do math” without using what they are used to thinking of as “math.”

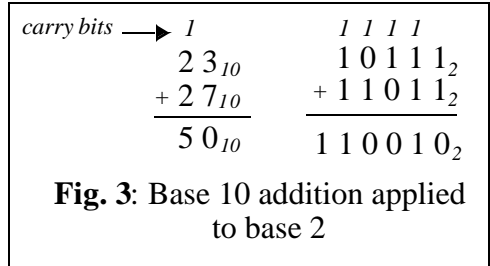


Fig. 3: Base 10 addition applied to base 2

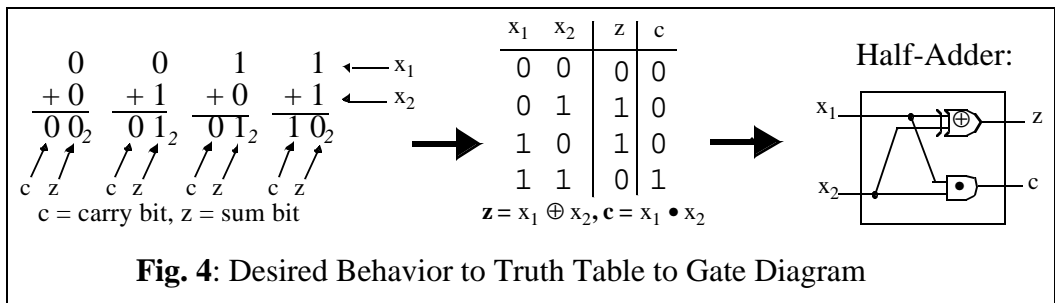


Fig. 4: Desired Behavior to Truth Table to Gate Diagram

STEP 3: Building a full-adder using half-adders, K-map simplification

Once we have designed the circuit (half-adder) to add two bits, we combine multiple half-adders into one large full-adder, with the slight modification of a carry-in bit to each half-adder. This allows us to add two multiple-digit binary numbers at a time, rather than just two bits at a time. Students are guided through the development of the truth-table for this, with the addition of a simplifying step called *K-maps*. The resulting sequence of 4 steps for circuit design are shown in Fig. 5.

1. Truth Table
 2. K-Map Simplification
 3. Function Notation
 4. Circuit Diagram
- Fig. 5:** Steps in creating a Circuit

STEP 4: Applying what we’ve learned

In this problem (Fig. 6 at right) we have a train switching yard with three switches (A, B, & C) leading off to three sidings (0, 1, & 2). Our job is to design the control unit circuit to set the three switches so that the train ends up on the correct selected destination track (0, 1, 2, or 3).

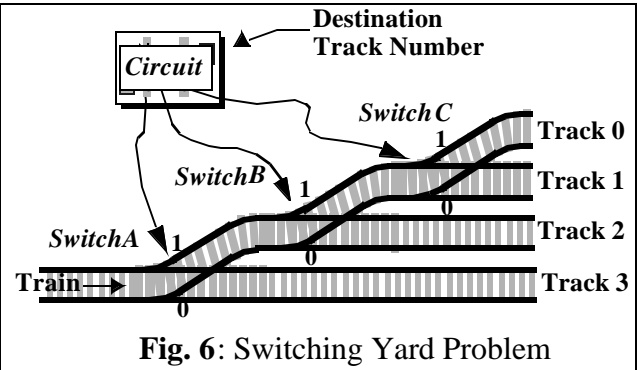


Fig. 6: Switching Yard Problem

For instance, if the destination track number is 1, then switch A = 1 (turn), switch B = 1, and switch C = 0 (straight).

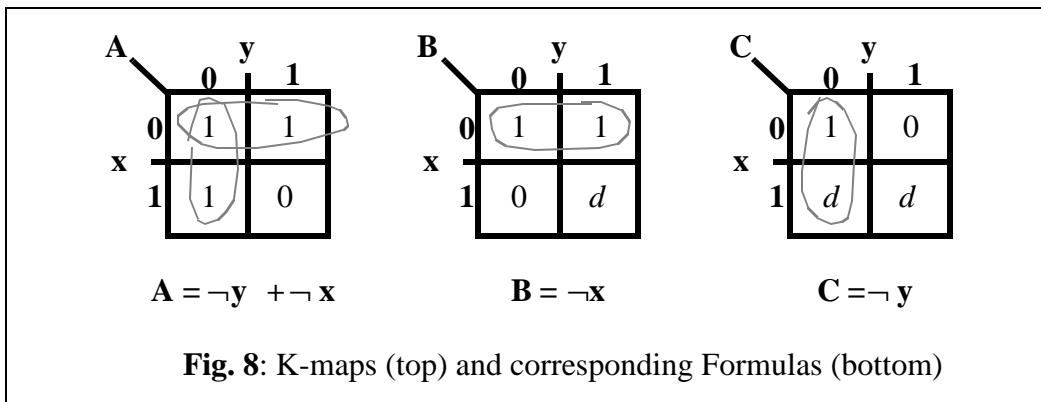
Observing the desired behavior of the train switches in Fig. 6 for each destination track, we first create the truth table shown in Fig. 7.

Next we rewrite the table entries for each of the outputs (A, B, & C) separately in a format called a

K-map (Fig. 8). This format allows us to group together outputs that have input in common. For instance in the top row of the K-map for switch A at left in Fig. 8, we group the two 1's since both of them have a *x* value of 0. These groups allow us to write the Formulas (bottom of Fig. 8) in a more abbreviated fashion than if we omitted K-map simplification.

Base 10	Base 2		Output: 1 = Turn, 0 = Straight, d = don't care		
	X	Y	Switch A	Switch B	Switch C
0	0	0	1	1	1
1	0	1	1	1	0
2	1	0	1	0	d
3	1	1	0	d	d

Fig. 7: Truth Table



Finally, the simplified circuit diagram shown in Fig. 9 is taken from the formulas. In this instance the unsimplified circuit would have used over twice as many gates (7 rather than 3).

Students' experience of understanding underlying technology gives them confidence in their later use of computers to do programming. They are also motivated to use the mathematical and engineering concepts of digital logic they learn in the hands-on laboratory setting to design and build circuits.

