MPRI : cours "Circuits synchrones"
Examen final : corrigé.

All circuits in this problem are *combinatorial* (i.e. memory-free). All circuits will be presented explicitly in two equivalent ways, by :

Schema a *Directed Acyclic Graph* of its structure, properly drawn from inputs to outputs, and documented by the symbolic names from the net-list.

Net-List The *net-list* $v_1 = e_1, \cdots v_n = e_n$ defines each variable v_i by an expression $e_i \in \{t \cap t', t \cup t', t \oplus t'\}$ in which terms t and t' represent : an input ; a constant 0,1 ; a *previously defined* variable, v_j with $j < i$.

1 Parity

The *parity* $p = \mathcal{P}_n(x)$ of a n-bit binary word $x = x_0 \cdots x_{n-1} \in \mathbf{B}^n$ is the word $p \in \mathbf{B}^n$ whose k-th bit is the sum modulo 2 of the previous bits of x :

$$p_k = x_0 \oplus \cdots \oplus x_k = \sum_{j \leq k} x_j \pmod 2. \tag{1}$$

Question 1 (Minimal size Parity circuit)
Describe the minimal size *parity circuit $\mathcal{P}_n$, with n bits of input $x_0 \cdots x_{n-1}$, and n bits of output $p_0 \cdots p_{n-1}$.*

1. *Show that output p_k in your circuit satisfies specification (1).*
2. *Show that it is* the unique minimal size *circuit for computing parity.*
3. *Analyze the combinatorial depth (maximal number of gates between inputs&outputs) of the circuit.*

Answer 1 *The net-list (schemas in fig. 1)*

$$v_0 = x_0 \text{ and } v_k = x_k \oplus v_{k-1} \text{ for } k > 0 \tag{2}$$

computes $v_n = \mathcal{P}_n(x)$ within $n-1$ xor gates (for $n > 0$).

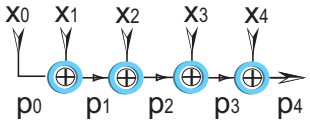

FIG. 1 – Minimal size parity $\mathcal{P}_5$.

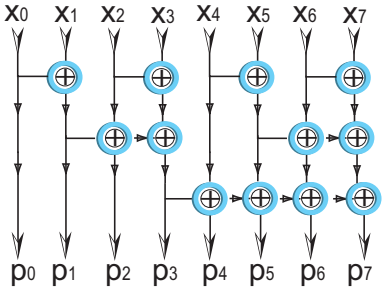

FIG. 2 – Minimal depth parity $\mathcal{P}_8$.

1. *Relation (1) follows simply by induction on* $k \in \mathbf{N}$.
2. *By induction, we prove that* n *is the minimal size for any parity circuit* $\mathcal{P}_{n+1}$*, and that (2) yields the unique such circuit. The case* $n = 0$ *is trivial. A parity circuit* $\mathcal{P}_{n+1}$ *over* $n+1$ *bits contains a* prefix *parity circuit* $\mathcal{P}_n$ *over its* n *least-significant bits. The outputs* $p_0 \cdots p_{n-1}$ *of the sub-circuit are all independent of input* x_n*. In a minimal size parity circuit, the prefix parity sub-circuit must also be optimal : otherwise the whole circuit would not be optimal. By induction, this sub-circuit is therefore unique and equal to (2), up to* p_{n-1}*. Output* $p_n = x_n \oplus p_{n-1}$ *depends on* x_n*, and one extra xor gate is therefore necessary and sufficient to compute* p_n.
3. *The combinatorial depth of parity* $\mathcal{P}_{n+1}$ *is equal to* n*, by (2).*

Question 2 (Minimal depth Parity circuit) *1. Show that the combinatorial depth* q *of any parity circuit* $\mathcal{P}_n$ *is bounded by* $q \geq \lceil \log_2(n) \rceil$.

2. *Construct a* minimal depth *parity circuit* $\mathcal{P}_n$*, for* $n = 2^q$.
3. *Analyze your circuit : number of gates, depth, wire area.*

Answer 2 *1. Bit* $p_{n-1} = \bigoplus_{j<n} x_j$ *depends upon the* n *input bits* $x_0 \cdots x_{n-1}$*. It follows that the depth* q *(number of unary/binary gates traversed through the net-list from some input to* p_{n-1}*) of this sub-circuit is such that* $2^q \geq n$*, otherwise one surely could not reach all* n *inputs. Applying logarithms in base 2 yields :* $p \geq \lceil \log_2(n) \rceil$.

2. *For $q = 0$, $n = 1$ and the (0 gate, 0 depth) circuit is simply $p_0 = x_0$. For $n = 2^q = m^2$, $q > 1$, we split the input in two parts $lx = x_0 \cdots x_{m-1}$ and $hx = x_m \cdots x_{n-1}$, of equal size $m = 2^{q-1}$. For $q > 0$, the circuit first computes recursively and in parallel, the parities $lp = \mathcal{P}_m(lx)$ and $hp = \mathcal{P}_m(hx)$ within depth $q - 1$. Both halves are then combined into full-length parities by $p_k = lp_k$ for $k < m$, and by $p_k = hp_{m-k} \oplus pm$ for $m \leq k < n$. where $pm = p_{m-1}$ represents the* middle parity*. The schemas for circuit $\mathcal{P}_8$ are shown in fig. 2.*
3. *The circuit has $qn/2$ xor gates ; its depth is q ; its wire area is qn.*

A SSA (Single Static Assignement) program is a list of instructions

$$[v_i = e_1, \cdots v_k = e_k].$$

Each instruction defines integer variable v_j by an expression

$$e_j \in \{t \oplus t', t \cup t', t \cap t', t + t', t - t'\},$$

in which t and t' represent terms which can be, either :

1. An input variable x.
2. An integer constant.
3. A previously defined variable v_i, with $i < j$.

For the purpose of this problem, the SSA program is executed on a 16 bits machine. Arithmetic operations are all computed modulo 2^{16}, at the rate of one instruction per cycle : the length k of the SSA code is equal to the program execution time (in cycles).

Question 3 (Optimal Parity software) *1. Provide an SSA program for computing $\mathcal{P}_{16}$. Analyze its length.*

2. *Endow the machine with a 16 bits* shifter*. Add the expression $z^i(t)$ to our SSA instructions. It expresses both up-shift $z^i(t) = 2^i t$ when $i \geq 0$ (expression $t << i$ in C), and down-shift $z^i = t \div 2^{-i}$ when $i < 0$ (expression $t >> -i$ in C). Provide a minimal length SSA program (including shifts) for computing the parity over 16 bits.*
3. *Remove the shifter again, and go back to question 3.1. Is your SSA code for $\mathcal{P}_{16}$ of minimal length ? If so, prove it ; else, provide a counter example with (say) less than 30 SSA instructions (without shift).*

Answer 3 *The generating series $x(z) = \sum_{k<n} x_k z^k$ and $p(z) = \sum_{k<n} x_k z^k$ are related by (1), hence $p(z) = \sum_k x(z) z^k = \frac{x(z)}{1-z} \pmod 2 \pmod{z^n}$.*

1. *An SSA program for computing the parity results from this expression :* $s_0 = x$, $q_0 = 0$, $s_{k+1} = s_k + s_k$, $q_{k+1} = q_k \oplus s_k$ *so that* $s_k = x(z)z^k$ *and* $q_k = \sum_{j<k} x(z)z^j$. *The parity* q_n *over* n *bits is computed in* $2n$ *SSA instruction ; so the SSA program for the parity over 16 bits has 32 instructions.*

2. *Expression*

$$\frac{1}{1-z} = \frac{1+z}{1-z^2} = \frac{(1+z)(1+z^2)}{1-z^4} = \frac{1}{1-z^{2^n}} \prod_{k<n} (1+z^{2^k}),$$

modulo z^{2^n} *is Euler's identity*

$$1 + z + \cdots + z^{2^n-1} = \prod_{k<n} (1+z^{2^k}),$$

which expresses the sum of 2^n *terms by an exponentially smaller product of* n *terms.*

There result an SSA code for a machine with shifts : $r_0 = x$ *and* $r_{k+1} = r_k \oplus z^{2^k} r_k$ *for* $k \in \mathbf{N}$. *The parity* r_n *over* 2^n *bits is computed in* $2n$ *instructions (the above expressions amount to 2 atomic instructions). For* $n = 16$, *the 8 atomic instructions program is :*

$$r_1 = x \oplus zx,\ r_2 = r_1 \oplus z^2 r_1,\ r_3 = r_2 \oplus z^4 r_2, ,\ r_4 = r_3 \oplus z^8 r_3 \qquad (3)$$

3. *While it can no longer be computed in one machine cycle, the shift operation* $s_i = z^{2^i} t$ *can be computed by the SSA sub-routine* $s_0 = t + t$ *and* $s_{k+1} = s_k + s_k$, *so that* $s_i = z^{2^i} t == 2^{2^i} t$ *is computed in* $i+1$ *cycles. Substituting into the previous SSA code (3) (with shifts) yields an SSA code (without shifts) of length* $2+3+4+5 = 14$ *instructions. Can one do with fewer instructions ? If you have a clue, let me know.*