Exercise 7: Spades of Gray

Task 1: Reflecting on Gray code

a) Given a B-bit Gray code, generate a $(B+1)$-bit Gray code by using the code twice, once forward with 0 as prefix and then reversed with 1 as prefix. Using this scheme recursively, write down a 3-bit *Binary Reflected Gray Code (BRGC).*

b) Expand the code to allow for metastability, by adding the "codewords" $G(x)*G(x+1)$ for $x \in [2^B]$, where G denotes the B-bit BRGCencoding function. We call the union of such strings and stable codewords *valid strings.* Order the valid strings such that the unstable strings stabilize only to adjacent stable codewords, where the order of the stable codewords is given by the natural order on the encoded values. Write the result down for your 3-bit code.

c) Define

$$\max_G\{G(x), G(y)\} \coloneqq G(\max\{x, y\})$$
$$\min_G\{G(x), G(y)\} \coloneqq G(\min\{x, y\}),$$

i.e., according to the order on stable BRGCstrings. Show that the metastable closure of $\max_G$ and $\min_G$ restricted to inputs that are valid strings is identical to what you would get if you extended $\max_G$ and $\min_G$ to valid strings $G(x) * G(x+1)$ according to the above order.

d) Prove that there are comparator circuits for valid strings w.r.t. the above order.

Solution

a) Let G and $\overline{G}$ denote the list of codewords of the B-bit Gray code and its reverse, respectively. Then the list of the $(B+1)$-bit code is given by $0G \circ 1\overline{G}$, where bG means to put $b \in \{0,1\}$ in front of each codeword and $\circ$ concatenation of lists. As G is a Gray code, the only place we need to check regarding the number of changing bits is when switching between the two lists. However, as the last codeword of G is the first of $\overline{G}$, the only differing bit is the front one, i.e., we indeed constructed a Gray code.

0	000	4	110
1	001	5	111
2	011	6	101
3	010	7	100

b) For a valid string $G(x) * G(x+1)$, $x \in [2^B]$, we order $G(x) <_G G(x) * G(x+1) <_G G(x+1)$. Given the already defined total order $<_G$ on stable strings, this results in a total order on all valid strings. As $G(x) * G(x+1)$ can stabilize only to $G(x)$ and $G(x+1)$, as $G(x)$ and $G(x+1)$ differ only in a single bit, this construction meets the requirements.

0	000	4	110
0-1	00M	4-5	11M
1	001	5	111
1-2	0M1	5-6	1M1
2	011	6	101
2-3	01M	6-7	10M
3	010	7	100
3-4	M10	---	---

c) Suppose s and t are valid strings with $s <_G t$. A simple, but tedious case distinction (looking at the table is the easiest way, though not very formal) shows that this implies that for any stabilizations s' of s and t' of t, $\max_G(s', t') = t'$ and $\min_G(s', t') = s'$. Thus, by definition of the metastable closure,

$$(\max_G)_{\mathrm{M}}(s, t) = \mathop{*}_{t \preceq t' \in \{0,1\}^B} t' = t$$
$$(\min_G)_{\mathrm{M}}(s, t) = \mathop{*}_{s \preceq s' \in \{0,1\}^B} s' = s\,.$$

d) By Theorem 6.6, we can implement $(\max_G)_{\mathrm{M}}(s, t)$ and $(\min_G)_{\mathrm{M}}(s, t)$. By c), these functions are consistent with the order on valid strings we defined above. Thus, we can implement comparator circuits for this order on valid strings.

Task 2: Conversion

a) Given circuits that convert B-bit BCRG and the reversed code to unary code, respectively, provide a circuit with $\mathcal{O}(2^B)$ additional gates that converts $(B + 1)$-bit BCRG to unary code.

b) Given a circuit that converts B-bit valid strings to $(2^B - 1)$-bit unary code (mapping $G(x) * G(x + 1)$ to $1^x \mathrm{M} 0^{2^B - 2 - x}$), provide a circuit with at most $2^B - 1$ additional gates and depth one larger that converts the reversed code.

c) Combine the previous results into a recursive construction that converts B-bit valid strings to $(2^B - 1)$-bit unary code with a circuit of size $\mathcal{O}(B2^B)$ and depth $\mathcal{O}(B)$.

d) Compare this result to what you would get from applying the construction from Theorem 6.6.

e*) Take a closer look at the circuits you constructed and/or the reverse code. Can you come up with a more efficient decoding circuit? By how much does size and depth improve in your new solutions?

Solution

a) Denote by C and $\overline{C}$ the circuits for the B-bit code and its reverse. Denoting by s the input, feed $s_{2\ldots B+1}$ into both circuits. The first $2^B - 1$ output bits are given by the bit-wise OR of the output of C with s_1. Bit 2^B equals s_1. The remaining $2^B - 1$ bits are given by the bit-wise AND of s_1 with the output of $\overline{C}$. This adds $2(2^B - 1)$ gates and increases depth by 1.

To see that this construction yields the correct output, observe that the structure of BCRG imposes that if $s_1 = 0$, the encoded number is given by decoding $s_{2\ldots B+1}$ as B-bit BCRG, which is guaranteed by the circuit. If $s_1 = 1$, the encoded number is $2^B + x$, where x is given by decoding $s_{2\ldots B+1}$ for the reverse code, which, again, is exactly what the circuit computes.

b) In the reverse code, the codewords "count down," i.e., the k^{th} codeword of the original code represents $2^B - k$. The circuit C decoding the original code outputs $1^{k-1}0^{2^B-k}$ for this codeword, but for the reverse code we need to output $1^{2^B-k}0^{k-1}$. By negating each output bit of C and reversing their order, this is easily achieved by adding $2^B - 1$ gates and increasing depth by 1.

c) Denote by $|C|$ the size of a circuit and by $d(C)$ its depth. Decoding a 1-bit code is trivial with a circuit of size 0 and depth 0. Given a circuit C_B for B-bit code, where $B \in \mathbb{N}$, we first apply b) to obtain a circuit $\overline{C}_B$ of size $|C_B| + 2^B - 1$ and depth $d(\overline{C}_B) = d(C_B) + 1$ decoding the reverse code. We then apply a) to obtain a circuit C_{B+1} decoding $(B+1)$-bit code of size $|C_B| + |\overline{C}_B| + 2(2^B - 1) = 2|C_B| + 3(2^B - 1)$ and depth $d(C_B) + 2$. Applying this construction inductively, we get a circuits C_B, $B \in \mathbb{N}$, with $|C_B| = \sum_{i=1}^{B} 2^{B-i} \cdot 3 \cdot (2^i - 1) < 3B2^B$ and depth $2(B-1)$.

It remains to show that the conversion is also working correctly in face of metastability, which we show by induction. This is trivial for a 1-bit code, so let's assume we have shown the claim for B-bit decoding circuits (and B-bit valid strings) and consider a $(B+1)$-bit valid string s fed into C_{B+1}. If $s_1 \neq \mathrm{M}$, observe that $s_{2\ldots B+1}$ is a valid string, so we can apply the induction hypothesis. Moreover, if $s_1 = 1$, the OR gates taking the output from C_B and s_1 as input guarantee that the first $2^B - 1$ output bits are stable 1. Similarly, if $s_1 = 0$, the AND gates ensure that the last $2^B - 1$ output bits are stable 0. Thus, the case $s_1 \neq \mathrm{M}$ is covered by the induction hypothesis.

Now consider the case $s_1 = \mathrm{M}$. In this case, $s_{2\ldots B+1} = 10\ldots0$, the codeword for $2^B - 1$ in a B-bit code. Hence, C_B and $\overline{C}_B$ output $1\ldots1$ and $0\ldots0$, respectively. Accordingly, the OR and AND gates ensure that all but the $(2^B)^{th}$ output bit are stable. Thus, the induction step succeeds.

d) We apply the construction from Lemma 6.5 and Theorem 6.6 to each output bit. For each bit we need a $\mathrm{MUX_M}$ with B select bits, which has size $\mathcal{O}(2^B)$ and (as each CMUX has constant depth and we use a binary tree of CMUXes of depth B) depth $\mathcal{O}(B)$. The total size of the circuit is thus $\mathcal{O}(B2^B)$ and its depth is $\mathcal{O}(B)$.

When checking the constants, however, we get depth 4 and size 5 (or 3 and 4, respectively, if we permit 3-input OR), so the construction here is slightly better. One should not take this too seriously, though, as both solutions can be further optimized by more clever use of (different) basic gates or optimizations on the transistor level. (That's why we usually think about asymptotic behavior when examining tasks in the level of detail we use in the lecture and exercises.)

Task 3*: Closure

a) Learn about Huffman's implementation of the metastable closure, based on prime implicants.

b) Show that implementing comparators for valid strings with the above order using his method results in very large circuits.

c) Decode what you learned in the TA session to the others!