國立清華大學
NATIONAL TSING HUA UNIVERSITY

EE 6250 VLSI Testing

Homework#2

資工系碩士班一年級

趙奕誠/ Yi-Cheng,Chao/105062600

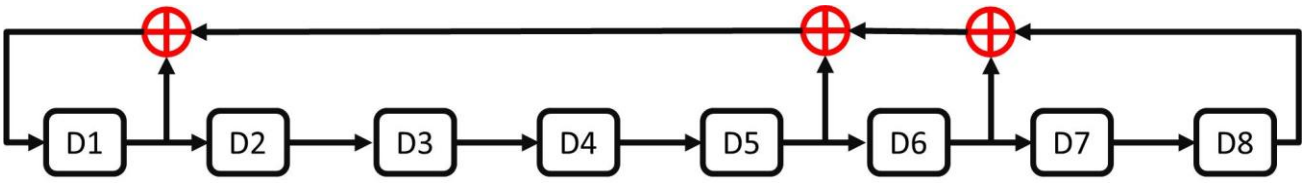
2016.12.22

VLSI Testing Homework #2

Yi-Cheng,Chao December 22, 2016

Answer

(a)



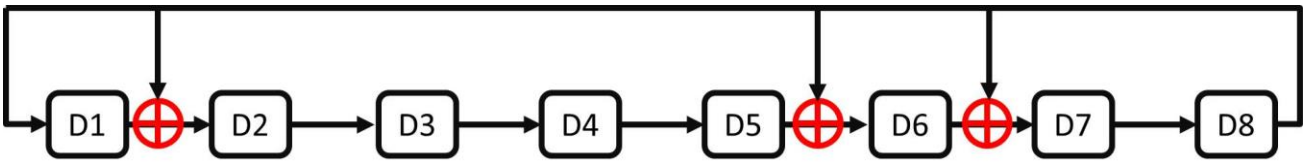
(b)

The execution result of my process as showed below:

```
0 0 0 0 0 0 1 0
0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0
count: 255
```

Because this 8-bit shift register can generate 255 states, which is a maximum length of 2^8-1, before the sequence repeats, it is a maximum-length generator.

(c)



(d)

By characteristic polynomial representing that the last bit (D8) will shift with no XOR operator to the first bit (D1), so there are eight times a sequence repeats, so total number of different patterns is 8.

Source Code (Enviroment: Linux)

主要含三個 function，以下逐一介紹功能：

(1)判斷 A 與 B 陣列是否相同，是的話回傳 true，反之則 false，型別使用 bool。

```
//compare A equal to B or not
bool compare(int *A, int *B){
    for(int i = 0; i < 8; i++){
        if(A[i] != B[i]) return false;
    }
    return true;
}
```

(2)使用一個 visit 陣列紀錄該輸入 pattern 是否出現過，若出現過則回傳 false，反之則將 visit 陣列中該 pattern 轉為十進位數後的位置紀錄為 true，並回傳 true。

```
//judge visit before or not
bool judge(int A, bool *V){
    if(V[A] == true) return false;
    else{//visit[A] hasn't visited before
        V[A] = true;
        return true;
    }
}
```

(3)將輸入的 pattern 由二進位轉為十進位，並回傳該十進位值。

```
// convert bin to dec
int bin_dec(int *A){
    int sum = 0;
    for(int i = 0; i < 8; i++){
        sum += (A[i] << i);
    }
    return sum;
}
```

主程式則做由特徵方程式產生之 companion matrix 的乘算，seed 使用 00000001 ($D_1D_2D_3D_4D_5D_6D_7D_8$)，並且由 do…while loop 來進行判斷是否要再進行迴圈，若遇到與起始 seed 相同的 pattern 則 break 出迴圈，但因為為了防止再出現與 seed 相同 pattern 之前便已出現兩組相同 pattern，因此我多用副程式 judge 內的 visit 來記錄該 pattern 出現過與否，主程式如下所示：

```
int main(){
    //companion matrix generated by characteristic polynomial
    int T[8][8] = {{ 1, 0, 0, 0, 1, 1, 0, 1},
                   { 1, 0, 0, 0, 0, 0, 0, 0},
                   { 0, 1, 0, 0, 0, 0, 0, 0},
                   { 0, 0, 1, 0, 0, 0, 0, 0},
                   { 0, 0, 0, 1, 0, 0, 0, 0},
                   { 0, 0, 0, 0, 1, 0, 0, 0},
                   { 0, 0, 0, 0, 0, 1, 0, 0},
                   { 0, 0, 0, 0, 0, 0, 1, 0}};

    //initial state
    int I[8] = { 0, 0, 0, 0, 0, 0, 0, 1};
    //seed
    int Q[8] = { 0, 0, 0, 0, 0, 0, 0, 1};

    int Q_next[8];
    bool *visit = new bool[256]();
    printf("D1 D2 D3 D4 D5 D6 D7 D8\n");
    printf("-----------------------\n");
    int count = 0;
    do{
        //Q+ = TQ
        for(int i = 0; i < 8; i++){
            int sum = 0;
            for(int j = 0; j < 8; j++) sum = sum ^ (T[i][j] * Q[j]);
            Q_next[i] = sum;
        }

        count++;
        for(int i = 0; i < 8; i++) printf(" %d ",Q_next[i]);
        printf("\n");

        //return true if Q_next if equal to I
        bool equal = compare(Q_next, I);
        if(equal == true) break;

        for(int i = 0; i < 8; i++) Q[i] = Q_next[i];

    }while(judge(bin_dec(Q_next) , visit));

    printf("count: %d\n", count);

}
```

Execution Result

程式會印出每次產生的 pattern 值並且顯示 pattern 的產生組數。

```
[105062600@pp02 VLSI_testing]$ g++ -o hw2 hw2.cpp
[105062600@pp02 VLSI_testing]$ ./hw2
D1 D2 D3 D4 D5 D6 D7 D8
-----------------------
 1  0  0  0  0  0  0  0
 1  1  0  0  0  0  0  0
 1  1  1  0  0  0  0  0
 1  1  1  1  0  0  0  0
 1  1  1  1  1  0  0  0
 0  1  1  1  1  1  0  0
 0  0  1  1  1  1  1  0
 0  0  0  1  1  1  1  1
 1  0  0  0  1  1  1  1
 0  1  0  0  0  1  1  1
 0  0  1  0  0  0  1  1
```

⋮

```
 1  1  0  1  1  1  1  0
 1  1  1  0  1  1  1  1
 0  1  1  1  0  1  1  1
 0  0  1  1  1  0  1  1
 0  0  0  1  1  1  0  1
 1  0  0  0  1  1  1  0
 1  1  0  0  0  1  1  1
 1  1  1  0  0  0  1  1
 0  1  1  1  0  0  0  1
 1  0  1  1  1  0  0  0
 0  1  0  1  1  1  0  0
 0  0  1  0  1  1  1  0
 0  0  0  1  0  1  1  1
 0  0  0  0  1  0  1  1
 0  0  0  0  0  1  0  1
 0  0  0  0  0  0  1  0
 0  0  0  0  0  0  0  1
count: 255
[105062600@pp02 VLSI_testing]$
```