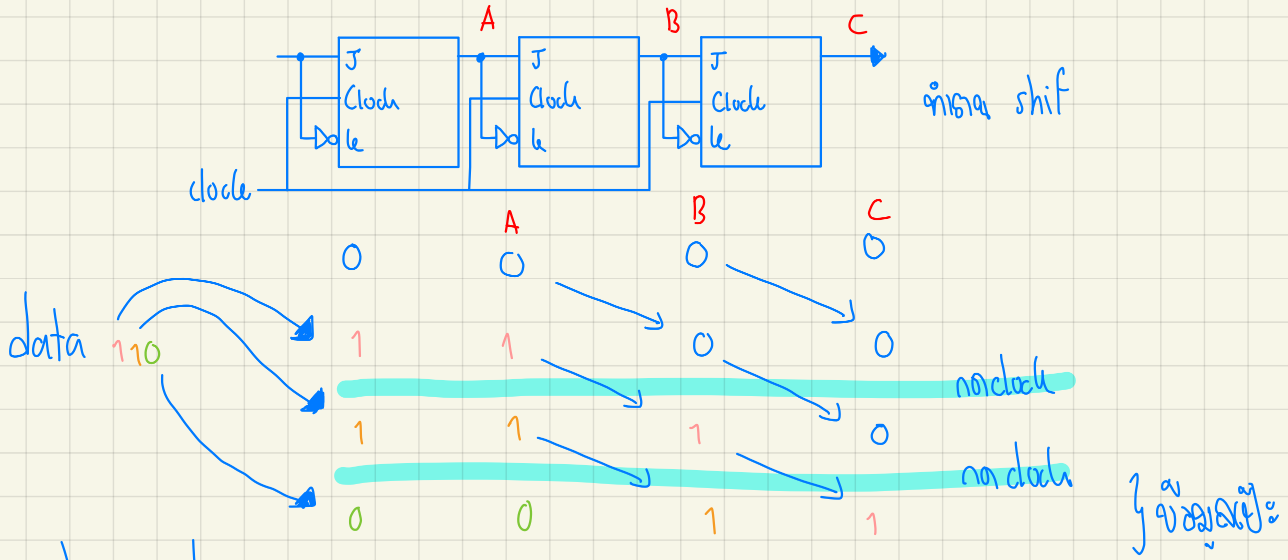
เมื่อเอา jk flipflop มาต่อกัน แบบ synchronous จะเป็นวงจรแบบ shift register



	A	B	C
0	0	0	0
1	1	0	0
1	1	1	0
0	0	1	1

cyclic code

↳ ตรวจก่อนเลยมี data word ที่ต้อง fix ขนาดเช่น C(28,7)

↳ ตอนนี้คือ data มีขนาดเท่าไหร่ก็ได้

↳ เอามาใส่ generator จะมองที่เราเรียกว่า divisor , หลักการเหมือนเดิม data+codeword ส่งไป

↳ G(x) divisor คือตัวหาร C(x) คือข้อมูลที่เราจะส่ง

R(x) remainder คือเศษจากการหาร

D(x) data คือข้อมูลที่เราจะส่ง

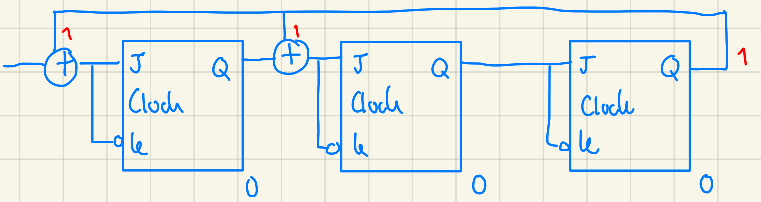
↳ ถ้า divisor n บิต เราจะใช้ FF n-1 ตัวในการทำงานด้าน hw

↳ $C(x) = x^3 D(x) \oplus R(x)$

↳ เราก็เอา XOR มาเข้าที่มีเลข 1

โจทย์ให้ divisor 1011 (x^3+x+1), data word 1001000 (D(x) x^3)

ใช้ FF ทั้งหมด 3 ตัว



สมมติว่า clock เข้าพร้อมกัน

ของแต่ละเลขจะเขียนผลลัพธ์ที่ Q

	0		0	0	
1	1		0	0	
0	0		1	0	
0	0		0	1	มีการ XOR เกิดขึ้น
1 → 1⊕1→0	0	0⊕1→1	1	0	
0	0		0	1	มีการ XOR อีกครั้ง
0 → 0⊕1→1	1	0⊕1→1	1	0	
0	0		1	1	หมดแล้ว

110 คือ remainder , ผลลัพธ์เหมือนเอาตัวหารยาวตามปกติเลย

กลับมาดูความสัมพันธ์ ที่ฝั่งรับ

รับ $C(x) = x^g D(x) \oplus R(x)$ ถ้าเอา $G(x)$ มาหาร → Divisor

$$Q(x) = \frac{x^g D(x)}{G(x)}$$

$$= \frac{x^g D(x)}{G(x)} \oplus \frac{R(x)}{G(x)}$$

$$= Q(x) \oplus \frac{R(x)}{G(x)}$$

เป็นขั้นตอนการเพิ่มเศษของตัวเองให้กับการหาร
→ เพื่อตอนรับจะนำไปเช็คว่าหารลงตัวไหม
→ ถ้าผิด 1 บิต มีโอกาสเปลี่ยน
→ จำนวนการแล้วมันเพี้ยน
→ กับการหารแล้วไม่ลงตัว เพราะเปลี่ยนแปลงเดิม

∴ ถ้าฝั่งรับนำ $C(x)$ ที่ได้มา หาร $G(x)$ เดียวกัน แล้วเศษเป็น 0 แสดงว่า ถูกต้อง

---

act กำหนด

$G(x) = x^3 + x + 1$ → divisor → 3 ตัว

Data = 1101010_2 → data

```
              0001111011        ← Q(x)
G(x) 1011 | 1101010000          ← x^g D(x)   (เติม 0 ต่อท้าย 3)
            1011
             1101
             1011
              1100
              1011
               1111
               1011
                1000
                1011
                 1100
                 1011
                  1110
                  1011
                   101          ← R(x)
```

ข้อ c กรณีข้อมูลผิด

```
          00000011010
1011 | 00011010101
          1011
           1100
           1011
            1111
            1011
             1000
             1011
              111
              000
              111   ← error
```

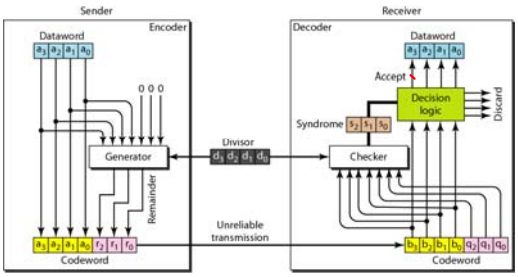
กรณีข้อมูลถูก

```
         00011111011
1011 | 11010101010 1
       1011
        1101
        1011
         1100
         1011
          1111
          1011
           1000
           1011
            1110
            1011
             1011
             1011
              000  → ถูกแล้ว
```

10.4 CYCLIC CODES

10.4.1 Cyclic Redundancy Check

Dataword	*Codeword*	*Dataword*	*Codeword*
0000	0000000	1000	1000101
0001	0001011	1001	1001110
0010	0010110	1010	1010011
0011	0011101	1011	1011000
0100	0100111	1100	1100010
0101	0101100	1101	1101001
0110	0110001	1110	1110100
0111	0111010	1111	1111111

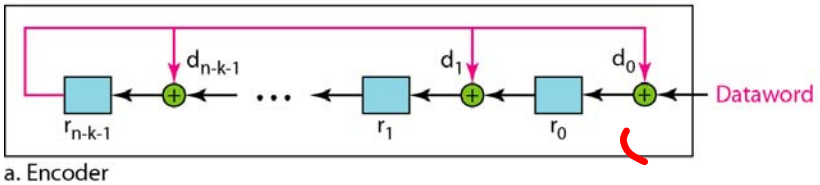


Division in CRC

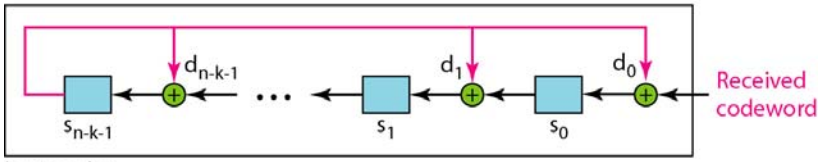
1) Encoder : Dataword 1001 / Divisor 1011

2) Decoder : Codeword 1001110 / Codeword 1000110

10.4.2 **Hardware Implementation** : Divisor, Augmented Dataword, Remainder



a. Encoder



b. Decoder

10.4.3 Polynomials

Degree of a Polynomial : $x^6 + x + 1$

Adding and Subtracting Polynomials : coefficients are only 0 and 1 => modulo-2

Multiplying or Dividing Terms / Shifting

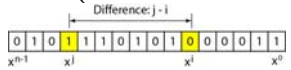
Cyclic Code Encoder Using Polynomials *

10.4.4 Cyclic Code Analysis

$$\frac{x^g D(x)}{G(x)} = Q(x) \oplus \frac{R(x)}{G(x)} \qquad C(x) = x^g D(x) \oplus R(x)$$

Single-Bit Error : If the generator has more than one term and the coefficient of x^0 is 1, all single errors can be caught.

Two Isolated Single-Bit Errors : If a generator cannot divide $x^t + 1$ (t between 0 and n - 1), then all isolated double errors can be detected.



Odd Numbers of Errors : A generator that contains a factor of x + 1 can detect all odd-numbered errors.

Burst Errors : ❑ All burst errors with $L \leq r$ will be detected.

❑ All burst errors with L = r + 1 will be detected with probability $1 - (1/2)^{r-1}$.

❑ All burst errors with L > r + 1 will be detected with probability $1 - (1/2)^{r}$.

A good polynomial generator needs to have the following characteristics:

1) It should have at least two terms.

2) The coefficient of the term x^0 should be 1.

3) It should not divide $x^t + 1$, for t between 2 and $n - 1$.

4) It should have the factor $x + 1$.

Standard Polynomials

Name	*Polynomial*	*Application*
CRC-8	$x^8 + x^2 + x + 1$	ATM header
CRC-10	$x^{10} + x^9 + x^5 + x^4 + x^2 + 1$	ATM AAL
CRC-16	$x^{16} + x^{12} + x^5 + 1$	HDLC
CRC-32	$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$	LANs

Advantages of Cyclic Codes

Other Cyclic Codes

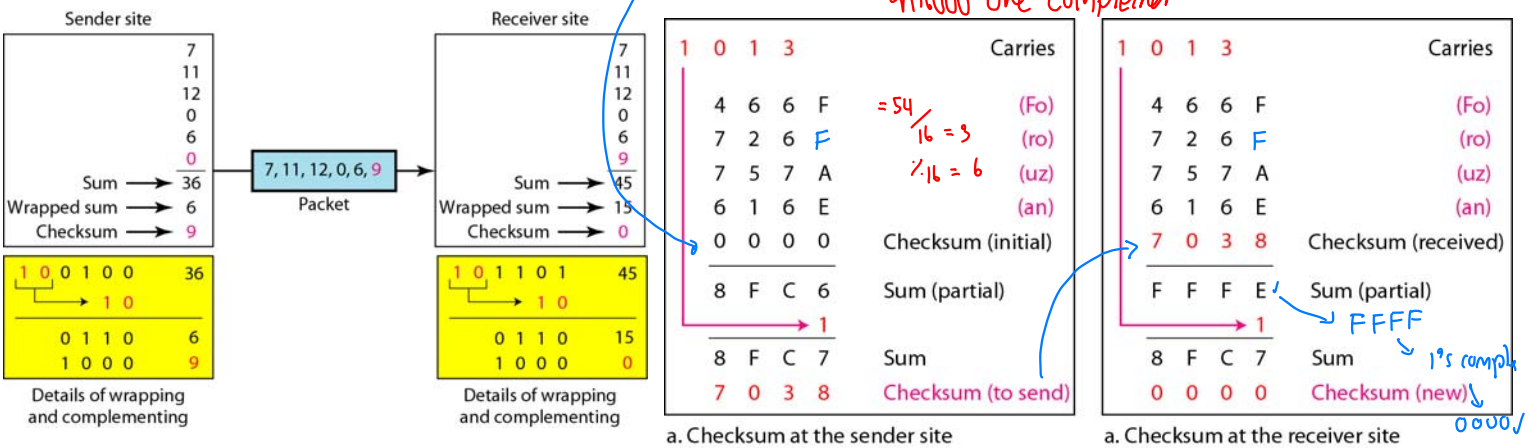
10.5 CHECKSUM

10.5.1 Idea

(7, 11, 12, 0, 6) => (7, 11, 12, 0, 6, 36)

(7, 11, 12, 0, 6) => (7, 11, 12, 0, 6, -36)

10.5.2 One's Complement



10.5.3 Internet Checksum

Sender site:	**Receiver site:**
1. The message is divided into 16-bit words. 2. The value of the checksum word is set to 0. 3. All words including the checksum are added using one's complement addition. 4. The sum is complemented and becomes the checksum. 5. The checksum is sent with the data.	1. The message (including checksum) is divided into 16-bit words. 2. All words are added using one's complement addition. 3. The sum is complemented and becomes the new checksum. 4. If the value of checksum is 0, the message is accepted; otherwise, it is rejected.