EUROPEAN ORGANISATION
FOR THE SAFETY OF AIR NAVIGATION
EUROCONTROL

CRC CALCULATION FOR MODE-S TRANSPONDERS

Task No. AT58
EEC Note N° 30/94

Approved for publication by
Head of Division B2
Issued : December 1994

REPORT DOCUMENTATION PAGE

Reference : EEC Note N° **30/94**	**Security Classification** : Unclassified
Originator Code : EEC Division B2	**Originator (Corporate Author) Name/Location** : EUROCONTROL Experimental Centre B. P. 15 F 91222 BRETIGNY SUR ORGE Cedex Telephone: 33 (1) 69 88 75 00 Fax : 33 (1) 60 85 15 04
Sponsor Code : EATCHIP Development Directorate	**Sponsor (Contract Authority) Name/Location** : EUROCONTROL Agency 72, rue de la Loi B 1040 BRUSSELS Telephone 32 (2) 729 35 11

Title : -

CRC CALCULATION FOR MODE-S TRANSPONDERS

Author : Mr. P. **HUNT**		**Pages**	**Figures**	**Tables**	**References**
		8	5		1

Det. Task Specification	**Period**	**Task No. Sponsor**	**Task No. Originator**
AT 58	November 1994	FCO.ET2.ST08	AT58

Distribution Statement :
(a) Controlled by : Head of Division B2
(b) Special limitations : None
(c) Sent to NTIS : YES ~~NO~~

Descriptors (keywords) : Mode-S Transponder CRC

Abstract :

During Mode-S Transponder tests, it was found that the software calculation of the CRC was taking too long and degrading the performance for other essential processors. This note describes the method used for implementing a faster CRC algorithm.

EEC Note N° **30/94**
EEC Task N° AT 58
Issued: December 1994

CRC CALCULATION FOR MODE-S TRANSPONDERS

Author Name : P. **HUNT**

EUROCONTROL Experimental Centre

Summary

During Mode-S Transponder tests, it was found that the software calculation of the CRC was taking too long and degrading the performance for other essential processes. This note describes the method used for implementing a faster CRC algorithm.

C O N T E N T S

1. GENERAL **1**

2. PARITY CHECK CODING IN MODE-S **2**

3. CALCULATION OF PI FOR A SQUITTER TYPE DF 11 ..... **4**

4. METHOD USED **5**

5. CONCLUSION **6**

6. REFERENCE **7**

1. GENERAL

In general the Cyclic Redundancy Check is made by hardware circuitry, as shown in ***Figures 2.9 and 2.10***. This is fast as it is calculated during transmission/reception of the signals and requires negligible extra time.

However, in order to verify the correct functioning of a transponder it is required to make an internal self test of the transmitted signals at regular intervals and to flag the transponder as unserviceable if this check fails.

The method used for the transponder is to verify the actual RF transmission signal against the expected signal.

The method used by the transponder in question is to generate a squitter interrogation and to count the number of low and high pulse transmissions in the outgoing pulse train and compare this with the expected number of transitions.

In order to do this, it is necessary to calculate the CRC pattern expected for the given pulse train. In this case, it is required to calculate the expected CRC by software in order to check the number of hardware transitions, all other data in the pulse train being known in the transponder.

As the transponder squitters an unsolicited DF11 interrogation at a random time interval between 800 and 1200 ms, the self test is performed on this transmission during transponder operation.

The squitter is used for the self test as it is an unsolicited transmission at regular intervals and is always active even when there is no other transponder activity.

2. PARITY CHECK CODING IN MODE-S

Parity check coding is used in Mode-S interrogations and replies to provide protection against errors. The Mode-S parity check code is defined in ICAO Annex 10 and recalled in paragraphs a), b) and c) below.

On receipt of a Mode-S interrogation, the Mode-S transponder shall perform a parity check, which is an examination of the sequence of demodulated bits to determine whether it is consistent with the code structure. If the bit sequence is consistent, the parity check is passed; otherwise it is failed. In this event, the interrogation shall not be accepted.

Similarly, the Mode-S transponder shall encode prior to transmitting a reply or squitter so that these transmissions are consistent with the code structure.

a) Parity Check Sequence :

A sequence of 24 parity check bits, generated by a code described in b) is incorporated into the field formed by the last 24 bits of all Mode-S transmissions. The 24 parity check bits are combined with either the address or the interrogation identification as described in c). The resulting combination then forms either the AP (Address / Parity) or the PI (Parity / Identification) field.

b) Parity Check Sequence Generation :

The sequence of 24 parity bits (P_1, P_2, ... P_{24}) is generated from the sequence of information bits (m_1, m_2,... m_k) where k is 32 or 88 for short or long transmissions respectively. This is done by means of a code generated by the polynomial :

$$G(x) = \sum_{i=0}^{24} g_i x^i$$

where g_i = 1 for i = 0, 3, 10 and 12 through 24 and is 0 otherwise.

When by the application of binary polynomial algebra the above G(x) is divided into $[M(x)]x^{24}$ where the information sequence M(x) is expressed as :

$$M(x) = m_k + m_{k-1} x + m_{k-2} x^2 + \ldots + m_1 x^{k-1}$$

the result is a quotient and a remainder R(x) of degree < 24. The bit sequence formed by this remainder shall be the parity check sequence. Parity bit p_i for any i from 1 to 24, is the coefficient of x^{24-i} in R(x).

Note : The effect of multiplying M(x) by x^{24} is to append 24 ZERO bits to the end of the sequence.

c) AP or PI Field Generation ::

The address used for AP field generation is either the discrete address or the All-Call address. The address shall be a sequence of 24 bits, (a_1, a_2 ... a_{24}). In the discrete address, a_1 shall be the bit transmitted first in the AA field of an All-Call reply. This address sequence shall be used in the downlink Address/Parity field generation, while a modified form of this sequence (b_1, b_2 ... b_{24}). shall be used for uplink Address/Parity field generation.

The interrogator identifier used for PI field generation is formed by a sequence of 24 bits (a_1, a_2 ... a_{24}).where the first 20 bits have zero value and the last four bits are a replica of the II field.

A modified sequence (b_1, b_2 ... b_{24}) shall be used for uplink AP field generation. Bit b_i is the coefficient of x^{i-1} in the polynomial H(x)A(x), where :

$$H(x) = 1 + x + x^2 + x^3 + x^4 + x^5 + x^6 + x^7 + x^8 + x^9 + x^{10} + x^{11} + x^{12} + x^{14} + x^{21} + x^{24}$$

and

$$A(x) = a_1 + a_2x + a_3x^2 + ... + a_{24}x^{23}$$

In the SSR Mode-S address, a_i shall be the i-th bit transmitted in the AA field of an all-call. In the all-call and broadcast adresses a_i shall equal one for all values of i.

The sequence bits transmitted in the AP or PI field is :

t_{k+1}, t_{k+2} ... t_{k+24}

The bits are numbered in order of transmission, starting with k+1.

In interrogations :

$$t_{k+i} = b_i \oplus P_i$$

where « ⊕« prescribes modulo-2 addition; i=1 is the first bit transmitted in the AP field.

In replies and squitters :

$$t_{k+i} = a_i \oplus P_i$$

where « ⊕« prescribes modulo-2 addition; i=1 is the first bit transmitted in the AP or PI field.

*Note : **Figures Nos. 2.9 and 2.10** show typical implementation of the error protection circuits.*

3. CALCULATION OF PI FOR A SQUITTER TYPE DF 11

It can be seen from Section 2 that the PI field to be calculated for a squitter transmission consists of the application of the encoding algorithm to the data stream consisting of the DF code, the CA field, the transponder Mode-S address and the interrogator identity 0.

Hence the data stream to be encoded is :

1			32	33 56
DF	0	CA	MODE-S ADDRESS	0 0

and the transmitted data shall be :

1			32	33 56
DF	0	CA	MODE-S ADDRESS	PI

where :

- PI is the Parity/Identification Field with II = 0,
- DF is the downlink Format DF = 11,
- CA is the capacity Field 0 to 3.

It can be seen that this particular case it is only required to pass the first 32 bits (1 to 32) through the encoding algorithm as the last 24 bits are all zero and will have no effect on the resulting PI generated.

4. METHOD USED

The method used is to divide the incoming data stream by the encoding polynomial defined for Mode-S CRC calculations.

This is done using modulo 2 addition as described in Section 2 which is equivalent to an exclusive OR function available in most computer languages.

The number of iterations required is equal to the number of bits in the incoming data stream.

Hence the execution time will depend on the number of bits in the data stream to be processed.

With long data streams methods exist where the parity may be calculated faster using look-up table(s) precalculated using the encoding polynomial as divisor but these methods are not described here.

Two examples are given of programmes to calculate a typical squitter CRC, one in PASCAL using 32 bit long word variables and the other in a MOTOROLA 6809 byte oriented machine in ASSEMBLER.

For information, the methods used for calculating the CRCs for the three normal cases used in Mode-S transmission are shown in Examples 3, 4 and 5. They are :

a) Sensor Encoder,

b) Transponder Decoder,

c) Transponder Encoder or Sensor Decoder

In these examples, it is assumed that :

- The coding polynomial is FFFA0480 hex G(x) of 2b),
- The message data is 20000000 hex,
- The A/C Mode-S address is DAB505 hex.

Hence the Sensor A/P field (uplink) will be 1347AC hex and the Transponder A/P field (downlink) will be 5AD35A hex.

5. CONCLUSION

The CRC calculation time in the transponder concerned was reduced from more than 40 ms to 2.4 ms, thus enabling the other processes to function normally in this real time environment.

This extremely simple method is fast enough for CRC calculations if the incoming stream is fairly short.

In any case, it is simple enough to make timing calculations depending on the number of iterations and to see if this method is suitable for a particular application.

6. REFERENCE

Minimum Operational Performance Standards for Air Traffic Control Radar Beacon System / Mode Select (ATCRBS / Mode-S) Airborne Equipment

Document No. RTCA/DO-181A
January 1992
Prepared by : SC-142

```
; CRC calculation for Mode-S transponder
; using mathmatical method
        DEFSEG RAM,ABSOLUTE,CLASS=DATA
        DEFSEG ROM,ABSOLUTE,CLASS=CODE
        SEG     ROM
        ORG     $100
*
START:
        LDA     #$80    ; initialise data
        STA     DAT     ; for test only
        CLR     DAT+1
        CLR     DAT+2
        CLR     DAT+3
        CLR     J       ; clear for bit count of 32
C001:   EQU     $
        LDD     DAT     ; Data bytes in
        BITA    #$80    ; test msb
        BEQ     C002    ; do nothing if zero
        EORA    P       ; else XOR with polynomial
        EORB    P+1
        STD     DAT
        LDD     DAT+2
        EORA    P+2
        EORB    P+3
        STD     DAT+2
C002:   EQU     $
        LSL     DAT+3   ; shift data left 1 bit
        ROL     DAT+2
        ROL     DAT+1
        ROL     DAT
*
        LDA     J
        INC     J
        CMPA    #31     ; 32 bits ?
        BLT     C001    ; no
* result in dat
        SEG     RAM
        ORG     $200
*
P       DB      $FF,$FA,$04,$80   ; Parity polynomial
DAT     DS      4                 ; Data stream 32 bits
*
J       DS      1                 ; 32 bit count
        END
```

Example 1

```
Program  CRC1 ;     { Mode-S CRC calculated by division }

{ This routine calculates the Downlink Parity Identification }
{ For a 56 bit Squitter Transmission }
{ This is done by hardware in the Sensor }
{ But verified by software in the auto-test procedure }

Uses Dos,Crt ;

VAR  j     : Integer ;

CONST
  poly : longint  = ($FFFA0480) ;  { Coding Polynomial }
     a : longint  = ($20000000) ;  { Uplink data }
    pi : longint  = ($00000000) ;  { Parity Identifier }

   BEGIN
   {  calculate Downlink Parity Identifier }
            pi := a  ;
         For j := 1 to 32 do begin
             If (pi AND $80000000) <> 0  then pi := pi XOR poly ;
                  pi := pi SHL 1  ;
                              end ;
   { result in pi }
   { total transmission = a,pi }
   END.
```

Example 2

```
Program  CRC4 ; { Mode-S ADDRESS  calculated by division }

{ This is the Transponder Encoder and Sensor Decoder }
{ This routine calculates the A/C Address or Address Parity }
{ This is done by hardware in the Transponder or Sensor }

Uses Dos,Crt ;


VAR  j         : Integer ;
     ad        : longint ;

CONST
  poly : longint  = ($FFFa0480) ;  { Coding Polynomial }
     a : longint  = ($20000000) ;  { Uplink data }
   adr : longint  = ($5ad35a00) ;  { A/C Address or AP }

   BEGIN

   { calculate the message parity  }

   For j := 1 to 32 do begin
             If (a AND $80000000) <> 0  then a := a XOR poly ;
    { Shift total message left }
                  a := a SHL 1  ;
                       end ;
    { Now XOR in the A/C address or Address Parity }

             a := a XOR adr ;
    { a = aircraft address or address parity }

   END.
```

Example 3

```
Program  CRC3 ;     { Mode-S ADDRESS  calculated by division }

{ This is the Transponder Decoder }
{ This routine calculates the A/C Address  }
{ This is done by hardware in the transponder  }

Uses Dos,Crt ;

VAR          j        : Integer ;
            ad       : longint ;

CONST
  p : longint  = ($FFFa0480) ;  { Coding Polynomial }
  a : longint  = ($20000000) ;  { Uplink data }
adr : longint  = ($1347ac00) ;  { Address Parity from Sensor }

BEGIN
{ calculate Aircraft Address  }
                  ad := 0 ;
                  { Treat Total Uplink Message }
                   For j := 1 to 56 do begin
           If (a AND $80000000) <> 0  then a := a XOR p ;
     { Shift total message left }
                   a := a SHL 1 + adr SHR 31 AND 1 ;
                   adr := adr SHL 1 ;
     { for the last 24 bits }
    If j > 31 then begin
               ad := ad + a SHR 31 AND 1 ;
               ad := ad SHL 1 ;
                   end ;
                                       end ;
            { adjust result in ad }
              ad := ad SHR 2 ;
END.
```

Example 4

```
Program  CRC ;     { Mode-S CRC calculated by division }

{ This routine is the Sensor Encoder      }
{ This is done by hardware in the Sensor }

Uses Dos,Crt ;

VAR  j                  : Integer ;
     par,b,ad,p         : longint ;

CONST
 poly : longint  = ($FFFA0480) ;  { Coding Polynomial }
    a : longint  = ($20000000) ;  { Uplink data 32 bits }
  adr : longint  = ($dab50500) ;  { Aircraft address 24 ms bits}

    BEGIN
                p := poly ;
     { 1st. calculate Uplink Message Parity }
     For j := 1 to 32 do begin
        If (a AND $80000000) <> 0 then a := a XOR poly ;
                  a := a SHL 1  ;
                       end ;
   { a = Uplink Message Parity }
   { Then encode the Mode-S address }
    For j := 1 to 24 do begin
             If (adr AND $80000000) <> 0  then b := b XOR p ;
                  p   :=  p  SHR 1  ;
                  adr := adr SHL 1  ;
             end ;
                 { b = Encoded Mode-S address 24 bits }
                 { Now XOR for the Uplink address parity }
                 par := a XOR b ;
  { par = Total Uplink Parity 24 bits }
  { Total Uplink Message = a,par 56 bits }
 END.
```

Example 5

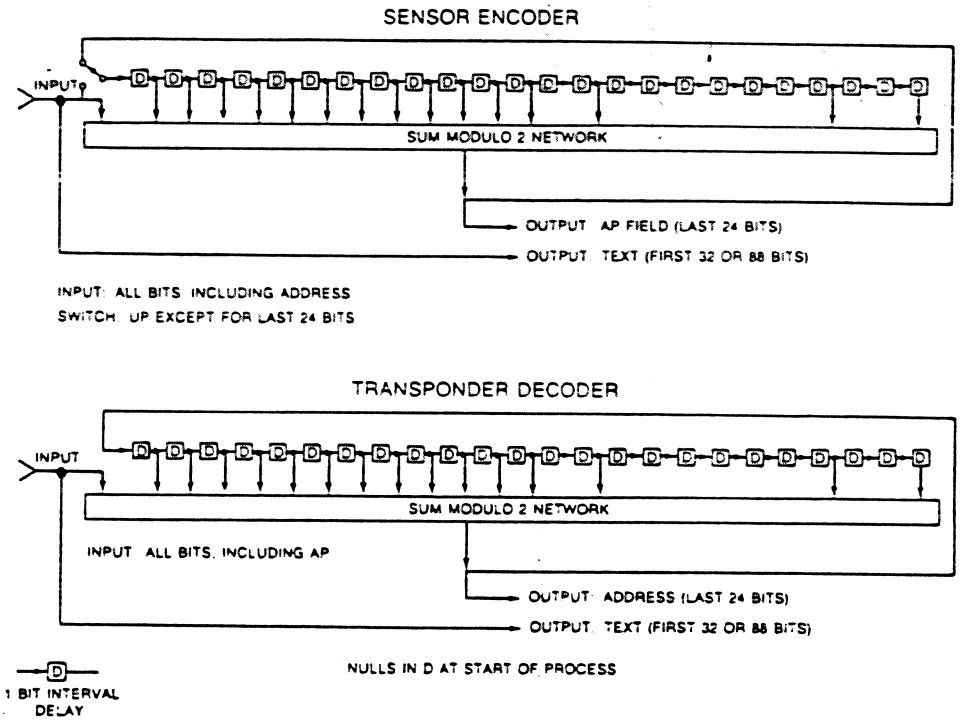
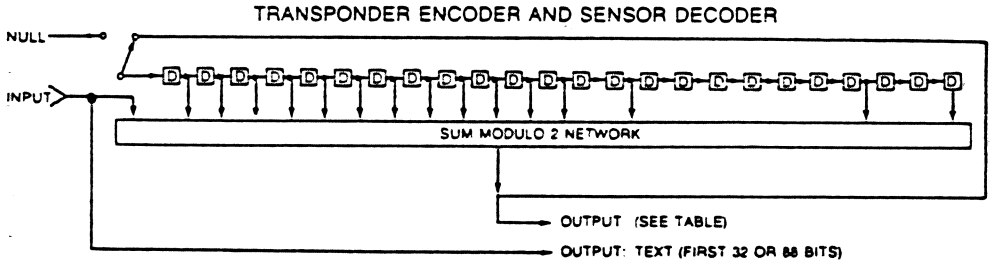


Figure 2-9 Functional Diagram of Uplink Coding



	TRANSPONDER	SENSOR
INPUT	ALL BITS, INCLUDING ADDRESS	ALL BITS INCLUDING AP FIELD
OUTPUT	AP FIELD = LAST 24 BITS	ADDRESS = LAST 24 BITS
SWITCH	RIGHT EXCEPT FOR LAST 24 BITS	

NULLS IN D AT START OF PROCESS

1 BIT INTERVAL DELAY

Figure 2-10 Functional Diagram of Downlink Coding