跳频通信系统信道编码的 DSP 实现

西安电子科技大学信息科学研究所(710071)　张光辉　李建东

摘　要：针对跳频通信对信道编码要求较高的问题，提出了混合纠错方式的 RS 码和重传反馈方式相级连的方法，主要研究了 RS 码的编码、解码过程以及 ARQ 协议的 DSP 实现。

关键词：Reed-Solomon(RS)　自动反馈重传(ARQ)　混合纠错编码(HEC)　信道编码

跳频通信是一种有效的抗干扰通信手段，具有良好的保密和抗干扰能力。跳频的频谱越宽，跳频越快，抗干扰能力也就越强，同时对信道编码的要求也就越高。因此为了保证跳频通信系统的可靠性，本文提出了混合纠错方式(HEC)即前向纠错方式(FEC)和重传反馈方式(ARQ)相级连的方法。前向纠错方式采用 RS 码，RS 码具有较强的纠错能力，特别适合纠正突发成片的错误，适用于突发信道；而自动反馈重传方式是自动要求重传协议，超出 RS 码的纠错范围，则自动反馈给发端要求重传。RS 码和 ARQ 相结合提高了信道的可靠性，从而保证了跳频通信的可靠性。

1　RS 码的编码和译码

给定任一有限域 $GF(p)$及其扩展域 $GF(q=p^m)$，其中 p 是素数，m 为某一正整数，则 RS 码是码长为 $n=q-1$ 的本原 BCH 码，一般取 $p=2$。RS 码可用(n,k,t)表示，其参数含义如下：码长 $n=2^m-1$，信息段为 k 个码元，监督段 $n-k=2t$，最小码距 $d=2t+1$，可纠 t 个码元错误，生成多项式为 $g(x)=(x+a)(x+a^2)(x+a^3)\cdots\cdots(x+a^{2t+1})$，其中 a 为 $GF(p^m)$上的本原元。显然，$g(x)$的全部根为 $a,a^2,a^3\cdots\cdots a^{2t}$，它的系数是 $GF(p^m)$的元素。由 $g(x)$生成的码是$(n,n-2t)$循环码，它包括那些能被 $g(x)$除尽且系数是 $GF(p^m)$的元素中次数不大于 $n-1$ 的多项式。RS 码的编译码是基于一组码元而不是单独的 0 或 1，这也是 RS 码纠错能力特别强的原因。这一特点使得 RS 码特别适合处理突发成片的错误。

1.1　编　码

RS 码编码可以生成系统码也可以生成非系统码。将所有码字都以多项式形式表示，对于 RS 码(n,k,t)，设生成多项式为：$g(x)=(x+a)(x+a^2)(x+a^3)\cdots\cdots(x+a^{2t+1})$

信息段多项式为：$m(x)=m_{k-1}x^{k-1}+m_{k-2}x^{k-2}+m_{k-3}x^{k-3}\cdots\cdots+m_1x+m_0$，其中 $m_l\in GF(p^m)$，$l=0,1\cdots\cdots k-1$

则系统码编码后的多项式为：$c(x)=m(x)x^{n-k}+m(x)x^{n-k} \bmod g(x)$

非系统码编码后的多项式为：$c(x)=m(x)g(x)$。

1.2　译　码

RS 码的系统码和非系统码的解码过程相同。设生成多项式为：

$g(x)=(x+a)(x+a^2)(x+a^3)\cdots\cdots(x+a^{2t+1})$，$a$ 为 $GF(p^m)$上的本原元，则 RS 码的校验矩阵 H 为：

$$H=\begin{bmatrix}(a^{m_0})^{n-1} & (a^{m_0})^{n-2} & \cdots\cdots & a^{m_0} & 1\\ (a^{m_0+1})^{n-1} & (a^{m_0+1})^{n-2} & \cdots\cdots & a^{m_0+1} & 1\\ \vdots & \vdots & \cdots\cdots & \vdots & \vdots\\ (a^{m_0+2t-1})^{n-1} & (a^{m_0+2t-1})^{n-2} & \cdots\cdots & a^{m_0+2t-1} & 1\end{bmatrix}$$

译码的主要步骤如下。

(1)计算伴随式

设 S^{T} 为伴随式矩阵的转置，H 为校验矩阵，R^{T} 为接收矩阵的转置，错误值为 $Y_i(i=1,2\cdots\cdots t)$，则：

$$S^{\mathrm{T}}=HR^{\mathrm{T}}=HE^{\mathrm{T}}=\begin{bmatrix}S_{m_0}=R(a^{m_0})=E(a^{m_0})\\ S_{m_0+1}=R(a^{m_0+1})=E(a^{m_0+1})\\ \vdots\\ S_{m_0+1+2t-1}=R(a^{m_0+2t-1})=E(a^{m_0+2t-1})\end{bmatrix}$$

(2)迭代计算错误位置多项式

迭代算法是 RS 解码的关键，也是最麻烦和最耗时的一步。设错误位置多项式为：$\sigma(x)=1+\sigma_1x+\sigma_2x^2\cdots\cdots+\sigma_tx^t$。

由初值 $\sigma^{(-1)}(x)=1$，$D(-1)=0$，$d_{-1}=1$，$\sigma^{(0)}(x)=1$，$D(0)=0$，$d_0=s_1$ 开始，j 从 0~$2t$ 循环执行以下语句：若 $d_j=0$，则 $\sigma^{(j+1)}(x)=\sigma^{(j)}(x)$，$D(j+1)=D(j)+1$，否则选 $\max(i-D(i))$，$d_i\neq0$ 的 i，且 $\sigma^{(j+1)}(x)=\sigma^{(j)}(x)-d_jd_i^{-1}x^{j-i}\sigma^{(i)}(x)$，最后得 $\sigma(x)=\sigma^{(2t+1)}(x)$。

(3)钱搜索计算错误位置多项式的根即接收码字的错误位置。

设 $R(x)=r_{n-1}x^{n-1}+r_{n-2}x^{n-2}\cdots\cdots+r_1x^1+r_0$，$\sigma(x)=1+\sigma_1x+\sigma_2x^2\cdots\cdots+\sigma_tx^t$ 为错误位置多项式，若 $\sigma(a^l)=0(l=1,2\cdots\cdots n)$，则 x^{n-1} 位置有错误，可得 $X_1=a_1^i\cdots\cdots$，$X_{e+t}=a_{e+t}^i$

(4)计算错误值

设实际产生的错误个数 $\gamma\leqslant t$，则令：

$\sigma_{j0}=\sigma_0=1$，$\sigma_{ji}=\sigma_i+\sigma_{j(i-1)}x_j$，$i=1,2\cdots\cdots\gamma-1$，可得：

$$Y_j=\frac{\sum_{i=0}^{\gamma-1}\sigma_{ji}s_{\gamma-i}}{x_j\sum_{i=0}^{\gamma-1}\sigma_{ji}x_j^{\gamma-i-1}},\ \hat{E}(x)=Y_1X_1^i+Y_2X_2^i\cdots\cdots+Y_{e+t}X_{e+t}^i$$

(5)根据错误位置和错误值对接收码字进行纠错

根据第(3)步和第(4)步确定的错误位置和错误值，在相应的 $R(x)$ 错误位置与错误值相加(此为域中相加，即为二进制的异或)。$\hat{C}(x)=R(x)-\hat{E}(x)$，纠错完毕送交上层处理。

2 ARQ 协议及其与 RS 的结合

ARQ 数据包中主要有 SN 和 RN，SN 表示发送端发送帧的序号，RN 表示收端接收到该帧后反馈给发端已接收到的序号。设发送端为 A，接收端为 B，A 到 B 的传送过程如下：

(1)在 A 端

①设 SN=0。

②从高层接收分组，并将 SN 赋予新的分组；如果没有新的分组到达，则等待，直到新的分组到达。

③传送带有 SN 的新分组。

④如果 A 接收到 B 发送给 A 的一个正确帧，且该帧中的 RN 大于 SN，则增加 SN，并返回到步骤②；如果在一定的时间内没有收到该正确帧，则返回到步骤③。

(2)在 B 端

①设 RN=0，并一直循环执行步骤②和步骤③。

②只要接收到从 A 发送到 B 的无误帧，且该帧中的 SN 等于 B 端的 RN，则将接收到的分组送到高层，同时 RN 加 1。

③在接收到该无误帧之后的一定时间内，传送包含 RN 的帧给 A。

节点 A 传送数据到节点 B 时发生的错误有二种：一种是 B 没有收到 A 发送的数据，另一种是 A 没有收到 B 发送的反馈。无论是第一种情况还是第二种情况，如果 A 在定时结束时没有收到 B 的反馈，或收到 B 的反馈，但收到 B 的 RN 并不大于 A 的 SN，则 A 会认为该组数据没有被 B 成功接收，于是重传该组数据帧。RS 码和 ARQ 相结合的具体帧格式如图 1 所示。其中，SN 和 RN 各占用 1 个字节。如果 RS 码采用 RS(31，20)，即 $n=31$，$k=20$，由 $n=2^m-1$ 得 $m=5$，即每个码元 5 位；又 $n-k=2t$，得 $t=5$，即可纠 31 个码元中的 5 个码元的错误。

SN	RN	INFORMATION	RS

图 1 RS 码和 ARQ 相结合的帧格式

对发送端由高层过来的分组先进行 ARQ 组帧，即把发送端本地的 SN 和 RN 赋值到帧中的 SN 和 RN 位置，同时启动定时器，进行 RS 编码。如定时结束时发端未收到收端的正确反馈，则发端要重传数据帧。若收到，则接收端首先进行 RS 解码，在解码的第②步迭代结束后，判断是否有 $\partial\circ\sigma(x)\leqslant t$(即错误位置多项式的最高次数是否超过 t，t 为 RS 的纠错码元数，此处 $t=5$)，如果 $\partial\circ\sigma(x)\leqslant t$，则继续进行 RS 解码，将解码后得到的数据帧进行判断，看是否有 RN>SN，以决定是否反馈给发端重传。如果 $\partial\circ\sigma(x)>t$，则将收端的 RN 保持不变直接反馈给发端，要求发端重传。

3 DSP 的实现与流程

DSP 采用 TMS320vc5410 作处理器。5410 有 3 个同步串口和 4 个外部中断。同步串口 1 与低层通信，同步串口 2 和 3 用来与高层通信。外部中断 1 提供 TDMA 同步定时信息，外部中断 2 和 3 与高层通信。采用模块化设计，3 个同步串口和 4 个外部中断各自采用中断模块。主程序又分为发送和接收二大模块，每个模块又按层次分为低层、RS、ARQ、高层。各模块之间及各层之间通过状态标志字来传递控制。具体的 RS 和 ARQ 内部还分为许多小模块，以 RS 码为例，其译码又分为计算伴随式、迭代、钱搜索、计算错误值和纠错模块。主要控制过程为同步串口 1 产生中断后，首先判断该时隙是发送还是接收，如果是发送时隙就判断发送模块的状态标志字来决定是否发送；若是则接收时隙调用接收模块，逐层执行各层的功能。具体的信道编码的 DSP 流程如图 2 所示。

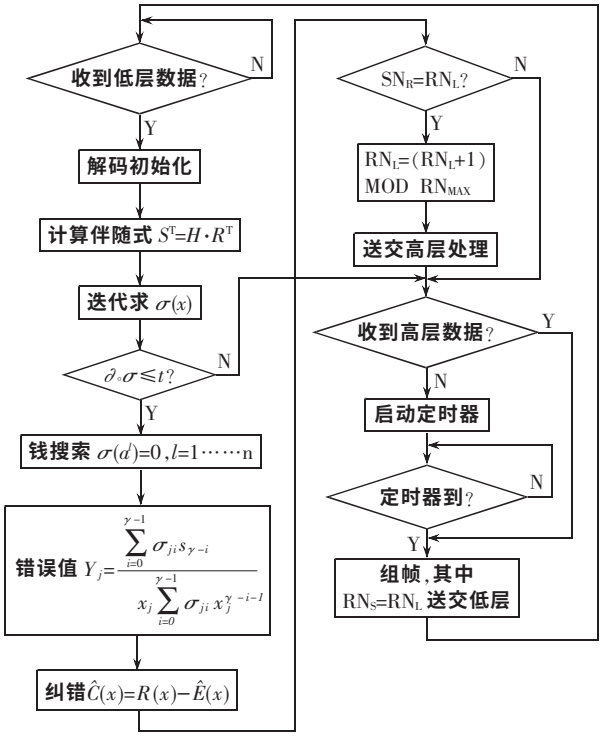

图 2　信道编码的 DSP 实现流程图

4　举　例

以本方法采用的 RS(31,20)为例,设:

T_{code}=01 11 08 04 00 00 0C 0C 1B 18 06 16 11 0D 0C 03 0B 03 05 1E 1C DF 1E 02 1F 16 00 07 00,而 R_{code}=01 11 08 04 00 00 0C 0C 1B 18 04 16 11 0D 0C 03 0B 03 05 1E 1C DF 1E 1F 16 00 0F 14

其中带下划线的为错误码元,显然有 3 个错误码元。

(1)计算伴随式

由 $S^T=HR^T$ 得 S=[2 1B 00 07 10 15 0D 0B 17 1F 15]。

(2)迭代计算错误位置多项式

具体的迭代算法如表 1 所示。

所以 $\sigma(x)=1+07x^2+06x^3$。

表 1　迭代算法求错误位置多项式

j	$\sigma^{(j)}(X)$	$D(j)$	$j-D(j)$	d_j
−1	1	0	−1	1
0	1	0	0	S_1=2
1	1+02x	1	0	1F
2	1+1Fx	1	1	01
3	$1+0Fx+12x^2$	2	1	18
4	$1+07x+17x^2$	2	2	05
5	$1+07x+12x^2+0cx^3$	3	2	1B
6	$1+07x^2+06x^3$	3	3	00
7	$1+07x^2+06x^3$	3	4	00
8	$1+07x^2+06x^3$	3	5	00
9	$1+07x^2+06x^3$	3	6	00
10	$1+07x^2+06x^3$	3	7	00
	$1+07x^2+06x^3$	3	8	00

(3)钱搜索计算错误位置多项式的根即为接收码字的错误位置

解 $\sigma(x)=1+07x^2+06x^3=0$ 得其根为 a^{19}、a^{30}、a^{31},因此错误位置为 x^{12}、x^1、x^0。

(4)计算错误值得错误多项式

由 $Y_j=\frac{\sum_{i=0}^{\gamma-1}\sigma_{ji}s_{\gamma-i}}{x_j\sum_{i=0}^{\gamma-1}\sigma_{ji}x_j^{\gamma-i-1}}$ 计算得在相应位置的相应错误值为 02、08、14,结合(3)得到的错误位置,进而得到 $\hat{E}(x)$。

(5)纠　错

$\hat{C}(x)=R(x)-\hat{E}(x)$=01 11 08 04 00 00 0C 0C 03 1B 18 06 16 11 0D 0C 03 0B 03 05 1E 1C DF 1E 02 1F 16 00 07 00

5　编译码实时性

本方法采用 TI 公司的 TMS320vc5410 的 DSP 处理芯片,其外接时钟为 12MHz,内部对其进行 8 倍频,则 CPU 时钟为 96MHz,即时钟周期为 1/96MHz=10.42ns。按 TMS320vc5410 执行 1 条指令的时间为 1 个时钟周期,则具体的编码和解码所执行的指令如下:

(1)编码:分割 722 条,合并 944 条,编码 4 218 条,一共 722+944+4 218=5884 条,耗时=5884 条×10.42ns/条=61.311μs。

(2)解码:分割 1 444 条,合并 472 条,初始化 237 条,伴随式 3 360 条,迭代 10 160 条,钱搜索 7 620 条,错误值 1 700 条,纠错 725 条,一共 25 718 条,耗时=25718 条×10.42ns/条=267.918μs。

编码共占用了 5 884 条指令,耗时为 61.311μs,解码共执行了 25 718 条指令,耗时为 267.918μs,在跳频保护时间内就可完成。对于实时性要求较高的跳频系统一般可满足要求。

通过实际的跳频电台的测试证明,这种 RS 和 ARQ 相结合的方式有效地保证了跳频通信的可靠性。

参考文献

1　王新梅,肖国镇.纠错码——原理与方法.西安:西安电子科技大学出版社,2001

2　Berttsekas D,Gallager R.Data Networks,Second Edition.Prentice Hall International Inc.,1992

3　陶德元,何小海,吴志华.RS 码编译码算法的实现.四川大学学报(自然科学版),2000;34(6)

4　张珣,罗汉文,宋文涛.RS 码在跳频系统中的应用及其实现.电信快报,2000;5(3)

5　李志勇,徐韦峰,周汀.基于 DSP 的 Reed_Solomon 编译码器的设计与实现.微电子学,2000;6(2)

(收稿日期:2003-08-25)