```
FUNCTION....: codec2_encode_450
                                                AUTHOR......: David Rowe

                                                DATE CREATED: Oct 1 2014
                                                Encodes 320 speech samples (40ms
                                              of speech) into 18 bits.
                                                The codec2 algorithm actually
                                              operates internally on 10ms (80
                                                sample) frames, so we run the
                                              encoding algorithm four times:
                                                frame 0: nothing
                                                frame 1: nothing
                                                frame 2: nothing
                                                frame 3: voicing bit, scalar Wo
                                              and E, 9 bit VQ of LSPs
                                                The bit allocation is:
                                                  Parameter
                                              frames 1-3   frame 4   Total
                                                  ------------------------------
                                              ------------------------------
                                                  Harmonic magnitudes (LSPs)
                                              0          9         9
                                                  Energy
                                              0          3         3
log Wo                                  0
5         5
                                                  Voicing
                                            0          1         1
                                                  TOTAL
                                            0         18        18

                                            \*-------------------------------------
                                            -------------------------------------*/
                                            void codec2_encode_450(struct CODEC2
                                            *c2, unsigned char *bits, short
                                            speech[])
                                            {
                                                   MODEL model;
                                                   float lsps[LPC_ORD];
                                                   float lsps_[LPC_ORD];
                                                   float ak[LPC_ORD + 1];
                                                   float e;
                                                   int lsp_indexes[LPC_ORD];
                                                   int Wo_index, e_index;
                                                   unsigned int nbit = 0;
                                                   assert(c2 != NULL);
```

```
memset(bits, '\0', ((codec2_bits_per_frame(c2) + 7) /
8));
                                                        /* frame 1 ----
                                                    ----------------------
                                                    ----------------------
                                                    -------- */
                                                        analyse_one_fra
                                                    me(c2, &model,
                                                    speech);
                                                        /* frame 2 ----
                                                    ----------------------
                                                    ----------------------
                                                    -------- */
                                                        analyse_one_fra
                                                    me(c2, &model,
                                                    &speech[N]);
                                                        /* frame 3 ----
                                                    ----------------------
                                                    ----------------------
                                                    -------- */
                                                        analyse_one_fra
                                                    me(c2, &model,
                                                    &speech[2 * N]);
                                                        /* frame 4: -
                                                    voicing, scalar Wo &
                                                    E, scalar LSPs -------
                                                    ---------- */
                                                        analyse_one_fra
                                                    me(c2, &model,
                                                    &speech[3 * N]);
                                                        pack(bits,
                                                    &nbit, model.voiced,
                                                    1);
                                                        Wo_index =
                                                    encode_log_Wo(model.Wo
                                                    , 5);
                                                        pack_natural_or
                                                    _gray(bits, &nbit,
                                                    Wo_index, 5, c2->gray)
e = speech_to_uq_lsps(lsps, ak, c2->Sn,
c2->w, LPC_ORD);
                                                        e_index = encode_energy(e, 3);
                                                        pack_natural_or_gray(bits,
                                                &nbit, e_index, 3, c2->gray);
                                                        encode_lsps_vq(lsp_indexes,
                                                lsps, lsps_, LPC_ORD);
```

```
                pack(bits, &nbit,
            lsp_indexes[0], lsp_pred_vq_bits(0));
                assert(nbit ==
            (unsigned)codec2_bits_per_frame(c2));
            }
            /*-----------------------------------
            -----------------------------------
            ---*\

              FUNCTION....: codec2_decode_450
              AUTHOR......: David Rowe
DATE CREATED: Oct 1 2014

              Decodes frames of 18 bits into 320 samples
            (40ms) of speech.
            \*-------------------------------------------
            ------------------------------*/
            void codec2_decode_450(struct CODEC2 *c2, short
            speech[],
                                   const unsigned char
            *bits)
            {
                MODEL model[4];
                int lsp_indexes[LPC_ORD];
                float lsps[4][LPC_ORD];
                int Wo_index, e_index;
                float e[4];
                float snr;
                float ak[4][LPC_ORD + 1];
                int i, j;
                unsigned int nbit = 0;
                float weight;
                COMP Aw[FFT_ENC];
assert(c2 != NULL);
                /* only need to zero these out due to (unused)
            snr calculation */
                for (i = 0; i < 4; i++)
                    for (j = 1; j <= MAX_AMP; j++)
                        model[i].A[j] = 0.0;
                /* unpack bits from channel ------------------
            ------------------ */
                model[3].voiced = unpack(bits, &nbit, 1);
                model[0].voiced = model[1].voiced =
            model[2].voiced = model[3].voiced;
                Wo_index = unpack_natural_or_gray(bits, &nbit,
            5, c2->gray);
                model[3].Wo = decode_log_Wo(Wo_index, 5);
```

```
model[3].L = PI /
model[3].Wo;
                                        e_index = unpack_natural_or_gray(bits,
                                &nbit, 3, c2->gray);
                                        e[3] = decode_energy(e_index, 3);
                                        lsp_indexes[0] = unpack(bits, &nbit,
                                lsp_pred_vq_bits(0));
                                        decode_lsps_vq(lsp_indexes, &lsps[3][0],
                                LPC_ORD, 1);
                                        check_lsp_order(&lsps[3][0], LPC_ORD);
                                        bw_expand_lsps(&lsps[3][0], LPC_ORD, 50.0,
                                100.0);
                                        /* interpolate ----------------------------
                                ------------------- */
                                        /* LSPs, Wo, and energy are sampled every
                                40ms so we interpolate
                                           the 3 frames in between, then recover
                                spectral amplitudes */
for (i = 0, weight =
0.25; i < 3; i++,
weight += 0.25) {
                                        interpolate_lsp_ver2(&lsps[i][0], c2-
                        >prev_lsps_dec,
                                                        &lsps[3][0], weight);
                                        interp_Wo2(&model[i], &c2->prev_model_dec,
                        &model[3], weight);
                                        e[i] = interp_energy2(c2->prev_e_dec, e[3],
                        weight);
                                }
                                for (i = 0; i < 4; i++) {
                                        codec2_lsp_to_lpc(&lsps[i][0], &ak[i][0],
                        LPC_ORD);
                                        aks_to_M2(c2->fft_fwd_cfg, &ak[i][0],
                        LPC_ORD, &model[i], e[i],
                                                &snr, 0, c2->lpc_pf, c2-
                        >bass_boost, c2->beta,
                                                c2->gamma, Aw);
                                        apply_lpc_correction(&model[i]);
                                        synthesise_one_frame(c2, &speech[N * i],
                        &model[i], Aw);
                                }
                                /* update memories for next frame ---------------
                        ------------ */
                                c2->prev_model_dec = model[3];
                                c2->prev_e_dec = e[3];
```

```
for (i = 0; i <
LPC_ORD; i++)
                                c2->prev_lsps_dec[i] = lsps[3][i];
                    }
                    /*---------------------------------------------------------------
                    --------------*\

                      FUNCTION....: synthesise_one_frame()
                      AUTHOR......: David Rowe
                      DATE CREATED: 23/8/2010
                      Synthesise 80 speech samples (10ms) from model parameters.
          \*--
          ----
          ----
          ----
          ----
          ----
          ----
          ----
          ----
          ----
          ----
          ----
          ----
          ----
          ----
          ----
          ----
          ----
          ----
          -*/
                  void synthesise_one_frame(struct CODEC2 *c2, short
                  speech[], MODEL * model,
                                          COMP Aw[])
                  {
                          int i;
                          PROFILE_VAR(phase_start, pf_start, synth_start);
                          PROFILE_SAMPLE(phase_start);
                          phase_synth_zero_order(c2->fft_fwd_cfg, model, &c2-
                  >ex_phase, Aw);
                          PROFILE_SAMPLE_AND_LOG(pf_start, phase_start, "
                  phase_synth");
codec2_postfilter(model, &c2-
>bg_est);
                                                  PROFILE_SAMPLE_AND_LOG(synth_start,
                                        pf_start, "    postfilter");
```

```
                synthesise(c2->fft_inv_cfg, c2->Sn_,
        model, c2->Pn, 1);
                PROFILE_SAMPLE_AND_LOG2(synth_start,
        "    synth");
                ear_protection(c2->Sn_, N);
                for (i = 0; i < N; i++) {
                        if (c2->Sn_[i] > 32767.0)
                                speech[i] = 32767;
                        else if (c2->Sn_[i] < -
        32767.0)
speech[i] = -
32767;
                else

                speech[i] = c2->Sn_[i];
                }
        }
```