```
// ============================================================================
// See https://www.teigfam.net/oyvind/home/technology/
//     253-my-audio-automatic-gain-control-sw-driver-agc-notes/#the_code
// Using only generic basic types, change from find_window_32ms_params_for_AGC
// ============================================================================

signed get_stn_from_nlz (const unsigned u_nlz, const unsigned nlz_lim_index) {
    return ((signed)u_nlz - (signed)nlz_lim_index);
} // get_stn_from_nlz

typedef struct {
    int32_t audio [NUM_MIC_SAMPLES_OF_SPECTRUM_512]; // 512 samples at 16 kHz for 32 ms
} AGC_bufs_t;

typedef struct {
    uint16_t   iof_audio_AGC;           // Range [0.. 511] 512 values frame_shifted when iof_audio_AGC==0
    AGC_bufs_t bufs [MUM_AGC_BUFS_2];   // Double buffer
    unsigned   iof_input_buf;           // 0 or 1
    unsigned   iof_output_buf;          // 1 or 0
    int32_t    audio_input_absval_mask; // Built from all samples in a buffer as "abs(signed_val)"
    unsigned   nlz_lim_index;           // nlz = num leading zeroes of the above (config 6 is 18 dB is s6_26)
    signed     stn_window_top;          // Previous shifted with value is window top
} AGC_t;

int32_t find_window_32ms_params_for_AGC_naked (
    const int32_t   audio_in,
    AGC_t           &AGC,                // AGC.iof_audio_AGC _must_ be incremented by 1 mod 512!
    signed          &s_stn_of_samples_io) // New s_stn_of_samples_io very NUM_MIC_SAMPLES_OF_SPECTRUM_512
{
    int32_t audio_out;
    AGC.bufs            [AGC.iof_input_buf] .audio[AGC.iof_audio_AGC] = audio_in;
    audio_out = AGC.bufs[AGC.iof_output_buf].audio[AGC.iof_audio_AGC];

    if (AGC.iof_audio_AGC == (NUM_MIC_SAMPLES_OF_SPECTRUM_512 - 1)) {
        AGC.audio_input_absval_mask or_eq (abs(audio_in)); // Final value

        const unsigned c_nlz_now                = get_NumLeadingZeros_asm_iron (AGC.audio_input_absval_mask);
        const signed   c_stn_new                = get_stn_from_nlz (c_nlz_now, AGC.nlz_lim_index);
        signed         stn_new_possibly_modified = c_stn_new;

        if (is_signed_in_window_lims_included (         // [-6 -5] or [4    5]
                AGC.stn_window_top - NLZ_FULL_WINDOW_1, // [low       [low
                c_stn_new,                              //
                AGC.stn_window_top)) {                  //    high]      high]
            // No code, keep c_stn_new as 0 (don't need stn_new_possibly_modified)

        } else if (c_stn_new < 0) { // (<0) c_stn_new is above AGC.nlz_lim_index: MUST scale down
            // No code, keep c_stn_new as < 0 (don't need stn_new_possibly_modified)

        } else { // c_stn_new > 0      (>0) c_stn_new is below AGC.nlz_lim_index: MAY scale up
            // May scale limited up <<, but not so much that noise from a falling leaf is heard
            stn_new_possibly_modified = min (c_stn_new, STN_MAX_GAIN_30DB_IN_6DB_STEPS);
        }

        AGC.stn_window_top = c_stn_new; // As explained above: don't return stn_new_possibly_modified
        s_stn_of_samples_io = stn_new_possibly_modified; // Possibly modified

        t_swap (unsigned, AGC.iof_input_buf, AGC.iof_output_buf); // Since MUM_AGC_BUFS_2 is 2
        AGC.iof_audio_AGC            = 0; // "frame_shifted"
        AGC.audio_input_absval_mask  = 0;

    } else {
        AGC.iof_audio_AGC++;
        AGC.audio_input_absval_mask or_eq (abs(audio_in));
    }
    return audio_out;
} // find_window_32ms_params_for_AGC_naked
  // Code Oyvind Teig as of v09520 of Beep_BRRR_02
```