`sim.asm` the 8088 assembly-language program that controls the hardware fading simulator

`ctabgen.for` computes scaled cosine tables for `sim.asm`

`ptabgen.for` computes phase increment tables for `sim.asm`

- program to compute BER and WER with various FEC codes

  `ceval.c` evaluates BER and WER performance of BCH or RS codes from the bit-error position information generated by `simrun.f`

- program to compute errors-per-word distribution

  `bkp.c` computes the distribution of the number of errors in a word using the independent error assumption from block bit error rates (BER_{block}) generated by `fdint.f`

- examples of awk and csh scripts for manipulating simulation program output

  `out2bers.csh` generates files containing block BERs for all SNR and block length combination by using the UNIX `awk` program.

  `out2bers.awk` an `awk` "script" to extract the block BERs for one SNR and one block size from the combined results in the simulation output file

  `out2runs.awk` an `awk` "script" to compute the distribution of the number of DFC passes from the simulation output file

`simmod.f` QAM encoding and decoding and OFDM modulation and demodulation

`simsnr.f` theoretical SN curve computation for an FM receiver, SN curve lookup, and baseband channel simulator

`simfec.f` BER and WER evaluation and simple FEC

`simut.f` utility subroutines and functions

`simhw.f` hardware interface and software preemphasis

`simdum.f` dummy hardware routines to replace simhw.f when compiling under UNIX

`sifft.dif` This file is the portion of the file FAST.FOR from [54] that follows subroutine FFA (with minor changes). It includes the routines FFA and FFS which use efficient (radix-8,4,2) FFT routines and that were used to implement OFDM (de)modulation. Since this is a large file and is widely available, the listing contains only the differences from the original as reported by the UNIX `diff` program.

- Monte-Carlo integration and BER bounds (these use many of the routines in the files described above)

  `fdint.f` Monte-Carlo integration

  `pint.f` BER bound computation

- analog interface routine

  `io.asm` is the 8088 assembly-language subroutine that reads/writes samples from/to the analog hardware interface

- fading simulator

Routine	Monte-Carlo Integration	Baseband Simulation	Hardware Measurement
data generator and bit/word error rate counters		•	•
Rayleigh fading	•	•	
channel simulation using SN curve look-up tables	•	•	
baseband noise generator		•	
QAM encoding/decoding and OFDM modulation/demodulation		•	•
pre-emphasis, phase synchronization, A/D and D/A			•

Table E.2: Routines used for BER evaluations.

`makeunix` UNIX makefile

`sim2ibm.sed` sed script to convert programs to the MS-DOS FORTRAN compiler

`sim2unix.sed` sed script to convert programs to the UNIX FORTRAN compiler

- baseband simulation and experimental measurements

`simrun.f` the main program for baseband simulations and experimental measurements

`simpdef.f` declarations of variables that control the simulations

`simpget.f` input of variables described in `simpdef.f`

`simfde.f` Rayleigh fading generator

`simdiv.f` diversity combining

`simgen.f` Gaussian random number generator and pseudo-random bit sequence (PRBS) generator

Appendix E

Program Listings

This appendix contains listings of the various programs mentioned in the thesis. Table E.1 shows the types of listings and the compilers used (the Unix version was SunOS release 3.2).

suffix	language	compiler(s) used
`.f`	FORTRAN 77	Unix `f77`, Microsoft FORTRAN 77 v.4.0.1
`.for`	FORTRAN 77	Microsoft FORTRAN 77 v.4.0.1
`.asm`	8088 assembler	Microsoft MASM assembler v.4.0
`.c`	C	Turbo C 2.0, `gcc`
`.csh`	`csh` shell	Unix `csh`
`.awk`	`awk`	Unix `awk`
`.sed`	`sed`	Unix `sed`, GNU `sed` (for MS-DOS)

Table E.1: Types of listings and compilers used.

Table E.2 shows the major groups of routines that are used by the different BER evaluation procedures. The table shows that none of the routines are common to all of the BER evaluation methods. This independence helps to prevent errors in one software module from affecting all of the results and makes it less likely that errors will go undetected.

The following list briefly describes the contents of the listings included in this appendix:

- makefiles and sed customization scripts

  `makefile` describes dependencies and how the various programs are compiled (for MS-DOS)

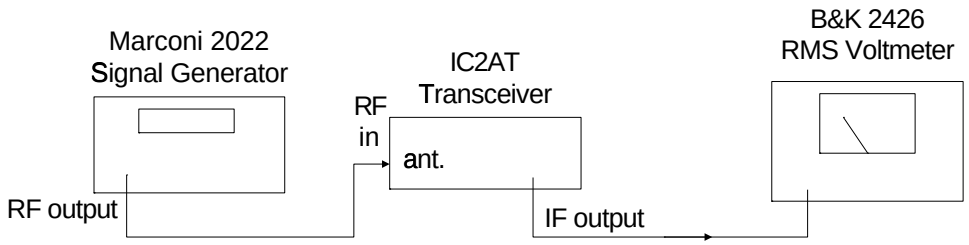

Figure D.4: Measurement of receiver noise bandwidth.

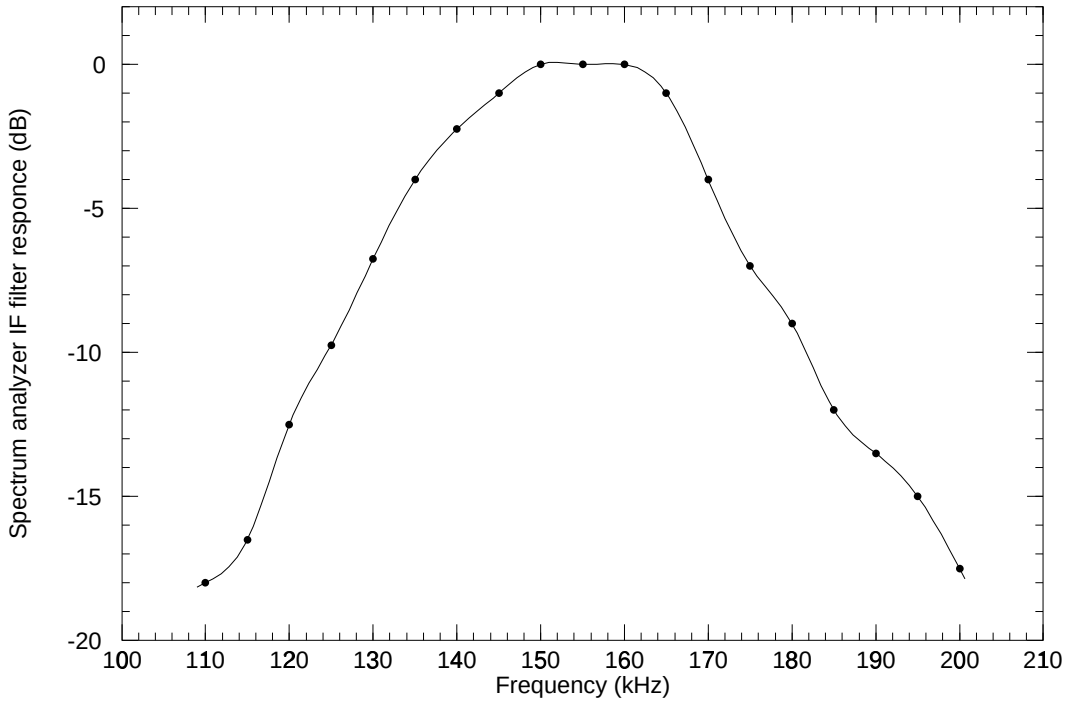

Figure D.3: Frequency response of spectrum analyzer 30 kHz filter.

D.4.2 Receiver IF Filter

Figure D.4 shows how the noise bandwidth of the receiver's IF filter was measured. The signal generator's frequency was varied and the signal at the output of the IF filter was measured with an RMS voltmeter. The signal level was kept within the linear region of the receiver's RF and IF amplifiers (42 mV RMS maximum at IF). A constant wide-band noise of 0.85 mV RMS was subtracted from all of the readings. The gain was measured over a 26 kHz range in steps of $\Delta f = 1$ kHz. The measurements are shown in Figure 5.7. Since the filter gain over the passband is not constant, the filter gain was calculated as the average gain over a 10 kHz range about the center frequency. The noise bandwidth was calculated to be 14.9 kHz.

measurement	value	units
s.a. noise bandwidth	34.4	kHz
receiver noise bandwidth	14.9	kHz
noise bandwidth correction	−3.6	dB

D.4.1 Spectrum Analyzer IF Filter

Figure D.2 shows how the measurement was made. The frequency response of the filter was measured by varying the frequency of the input signal and reading the signal level displayed by the spectrum analyzer. The frequency sweep was turned off (0 kHz/division) so that the analyzer remained tuned to one frequency. The gain of the spectrum analyzer's 30 kHz IF filter was measured over a 100 kHz range in steps of $\Delta f = 5$ kHz. The measurements are shown in Figure D.3. The noise bandwidth of the IF filter was found to be 34.4 kHz.

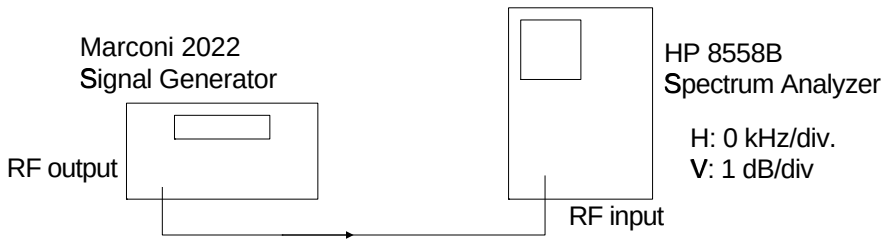

Figure D.2: Measurement of spectrum analyzer noise bandwidth.

The noise bandwidth of the IF filters used in HP spectrum analyzers is approximately 1.2 times the 3 dB bandwidth [82]. The nominal 3 dB bandwidth is the specified resolution bandwidth (30 kHz), giving a nominal noise bandwidth of about 36 kHz. The 3 dB bandwidth was measured to be 27 kHz, giving a noise bandwidth of about 32 kHz. Both estimates confirm the 34 kHz noise bandwidth measurement.

measurement	value	units
noise floor	−85.5	dBm
measured noise level	−81.0	dBm
correction for noise floor	−1.9	dB
detector and log amplifier correction	+2.5	dB
noise power	−80.4	dBm

D.4 Noise Bandwidth Measurements

The noise bandwidth, B_N, of a filter is defined as

$$B_N = \frac{1}{g_0} \int_{-\infty}^{\infty} g(f) df$$

where $g(f)$ is the power gain of the filter at frequency f and g_0 is the gain at filter's center frequency [49].

The noise bandwidths of the IF filters were calculated from the measured filter gain measurements with the approximation

$$B_N = \frac{\Delta f}{g_0} \sum_i g_i.$$

where the g_is are the filter gains measured at frequency intervals of Δf and g_0 is the filter gain in the passband (see Section D.4.2).

The noise bandwidth measurements of the spectrum analyzer and receiver IF filters are described in Sections D.4.1 and D.4.2 respectively. All of the SNR measurements were made using the spectrum analyzer's 30 kHz (resolution bandwidth) IF filter. The noise bandwidth of this filter was found to be 34.4 kHz. The noise bandwidth of the receiver's IF filter was found to be 14.9 kHz. The following table shows the correction required to compensate for the difference in IF noise bandwidths.

repeated several times over a period of several months and the same results were obtained to within ±0.5 dB. Note that the absolute power levels read by spectrum analyzer need not be accurate (and they were not) because the final SNR result is the difference of two powers measured in dB. The different noise bandwidth measurements in Section D.4 gave results within a few kHz. The SNR measurement accuracy is therefore probably between ± 1 and ± 2 dB.

D.3 Noise Power Measurement

The procedure for measuring RF noise power with a spectrum analyzer is explained in [82,83,84]. The noise power measurement requires two corrections.

The first correction is for the noise added by the spectrum analyzer's RF amplifier. The measured noise power is the sum of the spectrum analyzer's internally generated noise plus the noise of the device being measured. The first correction is therefore to subtract the noise power contribution of the spectrum analyzer from the measured noise power.

The second correction is for the response of the spectrum analyzer's envelope detector and log amplifier to non-sinusoidal signals. The output of the detector and log amplifier depends on the probability distribution of the received signal. The spectrum analyzer's display is calibrated for sinusoidal signals rather than Gaussian noise. The spectrum analyzer's noise power reading must be increased by 2.5 dB to correct for the effect of the noise distribution [82].

The following table shows the noise power calculation.

the impedance match at the power combiner[1]. The measured signal and noise levels are given below.

Since the noise is wide-band, the IF noise power measured by the spectrum analyzer will depend on the analyzer's IF filter bandwidth. Since the receiver has a narrower IF filter than the spectrum analyzer, the receiver's IF will receive less noise (relative to the signal) than the spectrum analyzer's IF. This effect can be corrected by scaling the noise power measurement by the ratio of the two IF filter noise bandwidths. Section D.4 describes the measurement of the noise bandwidths.

The calculation of the IF SNR (with 0dB signal attenuation) is as follows:

measurement	value	units
signal power	-9.1	dBm
noise power	-80.4	dBm
noise bandwidth correction	3.6	dB
computed IF SNR	74.9	dB

The attenuator can be used to reduce the RF signal level and thus reduce the SNR. For an SNR of X dB the attenuator is set to 75-X dB.

D.2 Accuracy

The accuracy of the SNR measurements is determined by the accuracy of the spectrum analyzer RF level measurements and the accuracy of the noise bandwidth measurements. The signal and noise level measurements were made by adjusting the spectrum analyzer's attenuators to bring the displayed level to a reference point. The accuracy specifications of the spectrum analyzer's step and vernier attenuators are $\pm$ 1 dB and $\pm$ 0.5 dB. The video filter was set to minimum bandwidth (about 1.5 Hz) to average the statistical fluctuations in the noise level. The vertical display was set to 1 dB/division. The measurements were

[1] Turning the noise generator off changes its output impedance.

Appendix D

RF SNR Measurement

D.1 RF SNR Measurement

As mentioned in Chapter 5, the RF SNR was measured instead of the IF SNR because the IF noise level could not be measured reliably. This appendix describes the RF SNR measurement.

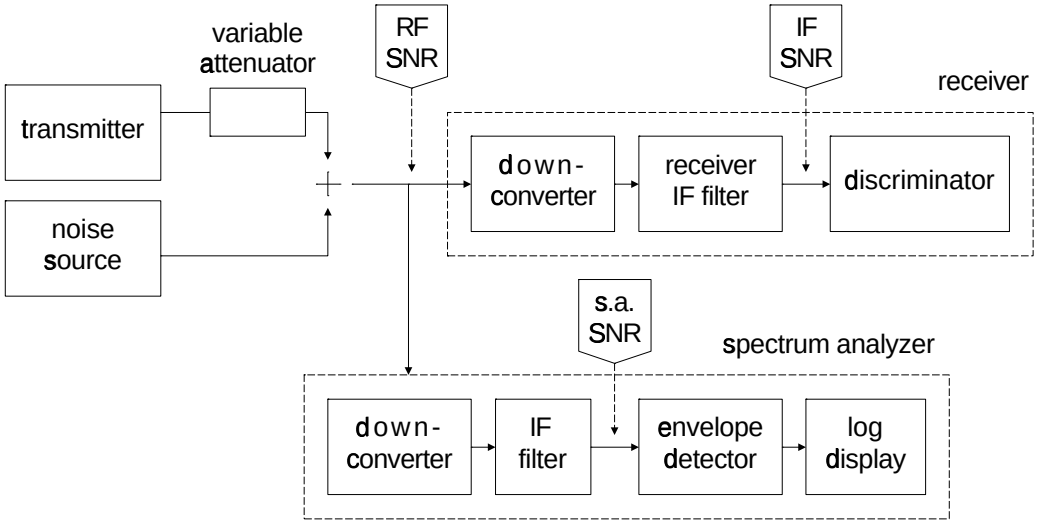

Figure D.1: Measurement of RF SNR.

Figure D.1 shows a block diagram of the equipment used. The signal and noise powers were measured with the spectrum analyzer. The RF signal power was measured after replacing the noise source with a 50 ohm termination. The noise power was measured after replacing the transmitter with a 50 ohm termination. The noise power measurement is described in detail in Section D.3. The 50 ohm terminations were used to maintain

C.4.4 Jitter

The A/D sampling clock jitter was checked by looking at the sample and hold signal, $\overline{S}$/H, with an oscilloscope. The delayed time base was set to display an edge of this signal at a sweep rate of 1 μs per division after a delay of 45 ms from the same edge. The cumulative peak jitter after the 360 clock periods was less than 1 μs, about 0.002 percent. Some or all of this may have been due to the oscilloscope's time base or noise on the triggering signal. The D/A sampling clock will have the same jitter because the D/A transfer signal is also controlled by the $\overline{S}$/H clock.

C.4.5 Clock Accuracy

The sampling rate was checked with a frequency counter. It was 8000 $\pm$ 1 Hz.

C.4.6 Overrun Indicator

The overrun indication was tested by increasing the sampling rate until overrun errors were obtained. The minimum reliable sampling period was 112 μs (8.9 kHz) on an 8 MHz IBM PC AT.

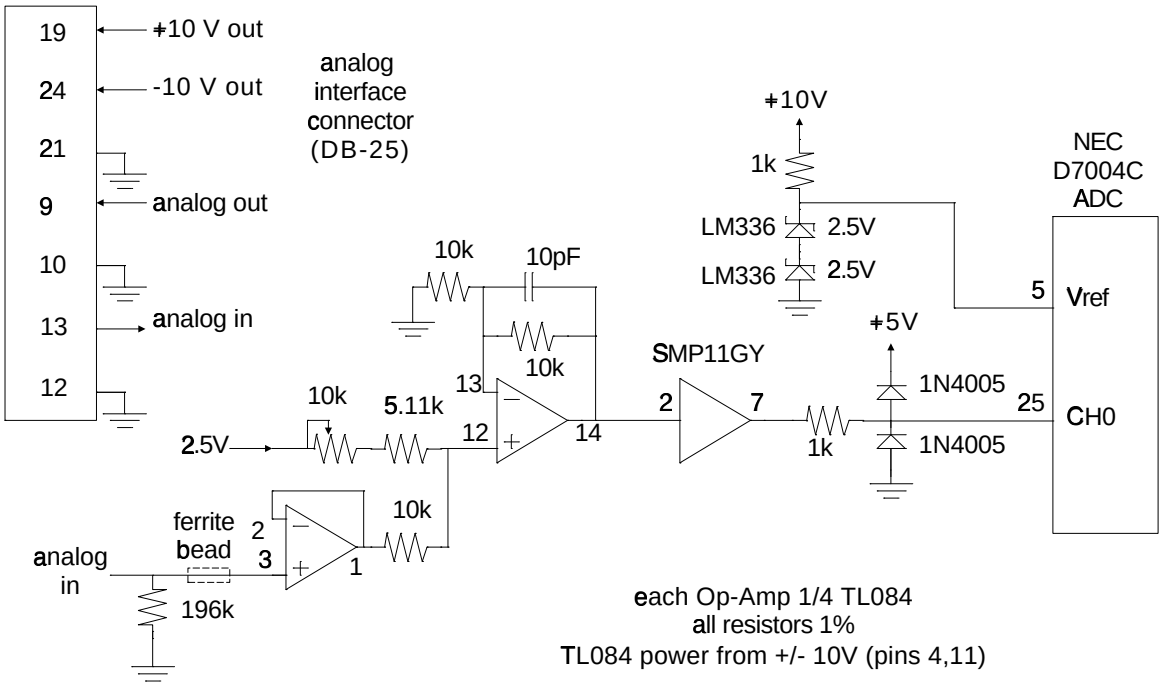

Figure C.6: A/D analog interface.

C.4.2 Distortion

Harmonic distortion was measured by sampling a 500 Hz, ± 2 volt sine wave produced by a signal generator and computing the power spectrum. The harmonic distortion was computed as the ratio of the power in a 100 Hz bandwidth about the sine wave frequency to the power outside that range. The ratio was over 45 dB.

C.4.3 Linearity and Accuracy

The linearity and range were checked by comparing the A/D and D/A readings to a DMM. The protection diodes were removed and 100,000 samples of a 0.1 Hz ramp function were taken. The values read from the A/D were checked and all possible values were present. The diodes limited the voltage at the A/D input to between 5.6 volts and −0.6 volts and the A/D sample values were restricted to between 474 and −512.

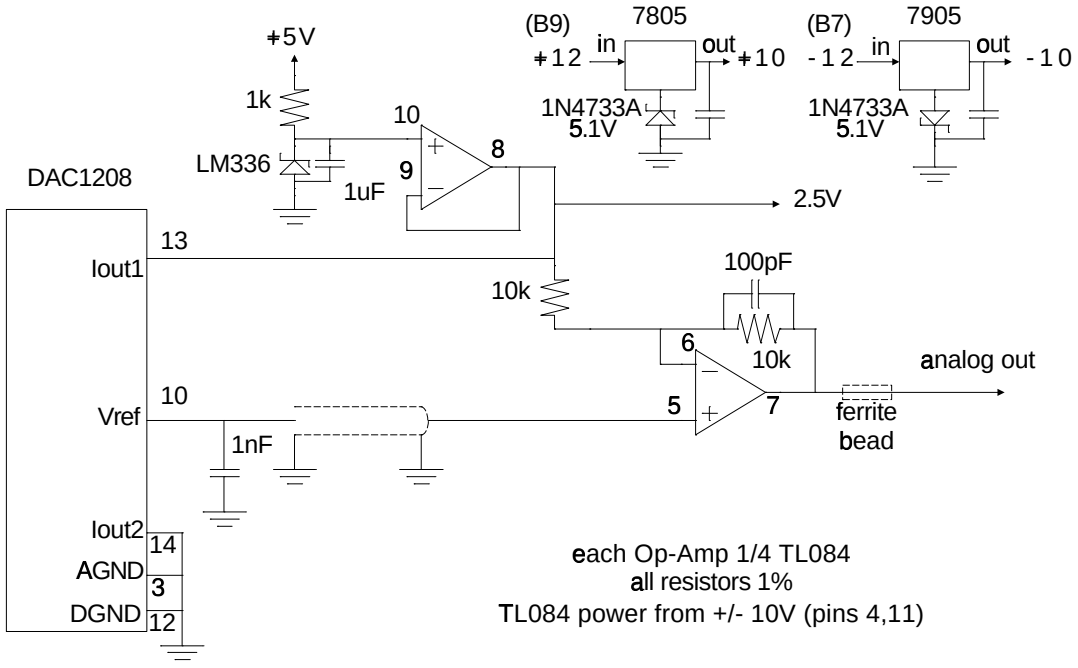

Figure C.5: D/A analog interface schematic.

The I/O subroutine turns off all maskable interrupts during sampling to avoid overruns that could occur while executing interrupt routines. This does not affect the DMA-based memory refresh.

C.4 Testing

C.4.1 Noise

Noise from the digital circuitry made the A/D reading fluctuate by about 1 sample value. One hundred thousand samples were taken with the A/D input disconnected (floating). About 0.1% were read as -1, about 0.2% were read as $+1$ and the rest were read as 0.

The measurement was repeated with the D/A output connected to the A/D input to check the noise on the D/A output. One sample was read as 0, about 66% were read as 1 and about 34% were read as 2.