Cascading several elements

A receiver is made up of three main elements: a preamplifier, a mixer, and an IF amplifier with noise figures of 3, 6, and 10 dB.

– If the overall gain of the receiver is 30 dB, and the IF amplifier gain is 10 dB,

What is the minimum gain of the preamplifier to achieve an overall noise figure of no more that 5 dB?

In [1]:
```julia
using Optim

# Given parameters in dB
NF_1_dB = 3
NF_2_dB = 6
NF_3_dB = 10
NF_total_max_dB = 5
G_IF_dB = 10
G_overall_dB = 30

# Convert dB to linear scale for noise figures and gains
NF_1 = 10^(NF_1_dB / 10)
NF_2 = 10^(NF_2_dB / 10)
NF_3 = 10^(NF_3_dB / 10)
NF_total_max = 10^(NF_total_max_dB / 10)
G_IF = 10^(G_IF_dB / 10)
G_overall = 10^(G_overall_dB / 10)

# Function to calculate total noise figure for given G1 in linear scale
function calculate_NF_total(G1_linear)
    G_1_G_2 = G_overall / G_IF  # Total gain divided by IF amplifier gain gives prod
    # Assuming G2 is fixed and we adjust G1, calculate total noise figure (linear sc
    NF_total = NF_1 + (NF_2 - 1)/G1_linear + (NF_3 - 1)/(G1_linear * G_1_G_2)
    return NF_total
end

# Objective function to minimize: difference between calculated NF_total and target
function objective_function(G1_linear)
    NF_total = calculate_NF_total(G1_linear)
    return (NF_total - NF_total_max)^2  # Squared difference for minimization
end

# Initial guess for G1_linear (since we don't have specific info, start with a reaso
initial_guess = 10  # Linear scale

# Use an optimization library to minimize the objective function and find optimal G1
result = optimize(objective_function, 1, 1000)  # Adjust bounds (1, 1000) as needed

# Extract the optimized G1 value
G1_optimized_linear = Optim.minimizer(result)
G1_optimized_dB = 10 * log10(G1_optimized_linear)

println("Optimized G1 in linear scale: ", G1_optimized_linear)
println("Optimized G1 in dB: ", G1_optimized_dB)
```

```
Optimized G1 in linear scale: 2.6315606874740136
Optimized G1 in dB: 4.202133899076101
```

– If its gain is set to this minimum,

What would the system noise figure become if the noise figure of the IF amplifier is increased to 20 dB?

In [2]:
```
# Define given parameters
NF_1_dB = 3  # Preamplifier noise figure in dB
NF_2_dB = 6  # Mixer noise figure in dB
NF_3_new_dB = 20  # Updated IF amplifier noise figure in dB
G_overall_dB = 30  # Overall system gain in dB
G_IF_dB = 10  # IF amplifier gain in dB

# Convert dB to linear scale for noise figures and gains
NF_1 = 10^(NF_1_dB / 10)
NF_2 = 10^(NF_2_dB / 10)
NF_3_new = 10^(NF_3_new_dB / 10)
G_overall = 10^(G_overall_dB / 10)
G_IF = 10^(G_IF_dB / 10)

# Placeholder for the optimized G1 value in linear scale from previous optimization
# Replace this with the actual value you found
G1_optimized_linear = 10  # This is a placeholder, replace with actual optimized G1

# Calculate G1 * G2 based on the overall gain and IF amplifier gain
G_1_G_2 = G_overall / G_IF

# Function to calculate the total noise figure with the updated NF_3
function calculate_total_noise_figure(NF_1, NF_2, NF_3_new, G1, G_1_G_2)
    # Total noise figure calculation using Friis formula
    NF_total = NF_1 + (NF_2 - 1) / G1 + (NF_3_new - 1) / (G1 * G_1_G_2)
    return 10 * log10(NF_total)  # Convert the total noise figure back to dB
end

# Calculate the new system noise figure with the updated IF amplifier noise figure
NF_total_new_dB = calculate_total_noise_figure(NF_1, NF_2, NF_3_new, G1_optimized_li

println("New system noise figure with updated IF amplifier noise figure: ", NF_total
```

```
New system noise figure with updated IF amplifier noise figure: 3.7882825438470857 d
B
```

Link Budget: Example

** Consider a GSM system with the following characteristics:

- Carrier frequency fc = 900MHz,
- Bandwidth B = 200kHz,
- Operating temperature T = 300 K,
- Antenna gains GTX = 8 dB and GRX = −2 dB,
- Cable losses at TX LTX = 2 dB,
- Receiver noise figure F = 7 dB.

** The propagation characteristics are

- The path loss exponent is n = 3.8,
- the breakpoint distance is 10 m,
- the fading margin is 10 dB.

The required operating SNR is 8 dB, the desired range of coverage 2 km.

What is the minimum TX power?

In [3]:
```
# Given parameters
𝑓꜀ = 900e6  # Carrier frequency in Hz
B = 200e3   # Bandwidth in Hz
T = 300     # Operating temperature in Kelvin
GTX_dB = 8  # Transmitter antenna gain in dB
GRX_dB = -2 # Receiver antenna gain in dB
LTX_dB = 2  # Cable losses at transmitter in dB
F_dB = 7    # Receiver noise figure in dB
n = 3.8     # Path loss exponent
d₀ = 10     # Breakpoint distance in meters
dᵩ = 2000   # Desired range of coverage in meters
Mf_dB = 10  # Fading margin in dB
SNR_req_dB = 8;  # Required operating SNR in dB
```

In [4]:
```
# Calculate path loss at the breakpoint distance (d₀) using the free-space path loss
PL_d0_dB = 20 * log10(d₀) + 20 * log10(𝑓꜀) - 147.55

# Calculate total path loss at the desired distance (dᵩ)
PL_dᵩ_dB = PL_d0_dB + 10 * n * log10(dᵩ / d₀)

# Calculate noise power in dBm
N_dBm = -174 + 10 * log10(B) + F_dB

# Estimate the minimum required transmit power in dBm
P_TX_min_dBm = SNR_req_dB + N_dBm + PL_dᵩ_dB + Mf_dB - GTX_dB - GRX_dB + LTX_dB

println("Minimum required TX power: ", P_TX_min_dBm, " dBm")
```

```
Minimum required TX power: 38.98428998065759 dBm
```

In [5]:
```
# Convert dBm to watts
P_W = 10 ^ ((P_TX_min_dBm - 30) / 10)

println("Minimum TX power in watts: ", P_W)
```

```
Minimum TX power in watts: 7.914600498295329
```

In [ ]:
```
```