```python
# Finding_Minima.py

import sympy as sp

def find_quadratic_minimum(a, b, c, sigma_a, sigma_b, sigma_c):
    # Define symbols
    x = sp.Symbol('x')

    # Define the quadratic function
    quadratic = a * x**2 + b * x + c

    # Compute the derivative
    derivative = sp.diff(quadratic, x)

    # Solve for critical point (minimum)
    critical_points = sp.solve(derivative, x)

    # Evaluate the quadratic at the critical point to find the minimum y value
    y_values = [quadratic.subs(x, cp) for cp in critical_points]

    # Determine the minimum y value and its corresponding x value (minimum point)
    min_y = min(y_values)
    min_x = critical_points[y_values.index(min_y)]

    # Uncertainty propagation
    sigma_y = sp.sqrt((sigma_a**2 + (sigma_b * min_x)**2 + (sigma_c * min_x**2)**2).evalf())
    sigma_x = sigma_b * (sigma_b * min_x)**2 / (2 * sp.sqrt(a * (a * min_x**2 + b * min_x + c)))

    return (min_x, min_y), (sigma_x, sigma_y)

# Example usage:
a = 6.3812e-11
b =  -4.9389e-09
c =  9.6456e-08
sigma_a = -2.7098e-09
sigma_b = 4.6798e-08
sigma_c = -2.2930e-09
```

```python

min_point, uncertainties = find_quadratic_minimum(a, b, c, sigma_a, sigma_b, sigma_c)
print("Minimum point (x, y):", min_point)
print("Uncertainties (delta_x, delta_y):", uncertainties)
```